IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 3, 2020, accepted November 24, 2020, date of publication November 30, 2020,
date of current version December 16, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3041393

Sandsbots: Robots That Sync and Swarm

AGATA BARCIŚ[1] AND CHRISTIAN BETTSTETTER[2], (Senior Member, IEEE)
[1]Karl Popper Kolleg on Networked Autonomous Aerial Vehicles, University of Klagenfurt, 9020 Klagenfurt, Austria
[2]Institute of Networked and Embedded Systems, University of Klagenfurt, 9020 Klagenfurt, Austria

Corresponding author: Agata Barciś (agata.barcis@aau.at)

This work was supported in part by the Karl Popper Kolleg on Networked Autonomous Aerial Vehicles, University of Klagenfurt, and in part by the Austrian Science Fund under Grant P30012.

ABSTRACT This article presents a multi-robot system that forms emergent space-time patterns. Inspired by the theory of swarmalators, in which synchronization and swarming of agents are mutually coupled, we propose a robot-suitable model for coordination in time and space. The approach is evaluated by simulations and demonstrated as proof of concept using small robots and drones. The novel building blocks comprise a time-discrete swarm aggregation model — which works robustly with low update rates in systems with communication delays — and specific functions that couple this spatial model to a discrete temporal coordination model, resulting in an overall discrete spatio-temporal coordination model.

INDEX TERMS Emergence, mobile robots, multi-agent systems, multi-robot systems, self-organization, swarming, synchronization.

I. INTRODUCTION

Multiple robots performing a joint mission must coordinate their behavior in time and space. Temporal coordination is needed for robots to act in synchrony, e.g., to lift an object or take photos simultaneously from different points of view. Sometimes it is necessary to avoid synchrony rather than achieving it, e.g., if mobile robots share a charging station. Spatial coordination is required for mobile robots to avoid collisions, form patterns in space, assemble, or spread out for coverage.

Algorithms for coordination in multi-robot systems have been proposed in a variety of applications on different functional levels and using different approaches. One stream of research intends to adapt approaches from self-organizing systems found in nature. In this domain, synchronization and other forms of temporal coordination are often modeled using the theory of coupled oscillators [1], and spatial coordination often relies on swarming and flocking [2]. These biologically-inspired algorithms are typically adaptive to changes, robust against failures of single agents, and scalable with the cardinality of the system.

The key motivation of our research is that temporal and spatial coordination have largely been treated independently so far [3], both in theory and practice. Mobile robots would, however, benefit from fusing the two to a unified model. For example, robots could not only take photos of a point of interest simultaneously but also form a spatial pattern around this point and take a sequence of photos sorted by the viewing angle in a self-organizing manner. A mathematical model for such a fusion, more specifically a *bi-directional coupling* of synchronization and swarming, was recently introduced by O'Keeffe, Hong, and Strogatz [3]. That article lays down the theoretical foundations of agents called "swarmalators" (a neologism for swarming oscillators), but the model itself is unsuited for mobile robots due to some assumptions that do not hold for robotics, such as time-continuous coupling between agents, an infinitely small agent size (no collisions), and unconstrained movement mechanics. This is why we adapted the original model and presented an initial proof of concept to demonstrate the potential of swarmalators for technical systems [4]. During the experiments we ran into several problems related to constraints in robot mechanics and wireless communications as well as the impact of delays (see Appendix). These hurdles motivated us to further pursue this path. Based on the lessons learned, the article at hand presents the overall result. The robots acting according to our approach are called "sandsbots" (a neologism for synchronizing and swarming robots).

The main contribution of this work is the model suitable for multi-robot systems that allows to form emergent space-time patterns. It consists of two mutually coupled parts: temporal coordination [5] and a novel swarm aggregation approach with time-discrete interactions between agents. The proposed model displays the following properties:

The associate editor coordinating the review of this manuscript and approving it for publication was Yang Tang.

- Has low communication requirements and guarantees robust behavior in the presence of communication delays and with a very low frequency of interactions between agents (i.e., once each few seconds).
- Adapts the maximum speed of agents to the communication rate, allowing the agents to avoid collisions and successfully form a pattern.

We verify the sandsbot model in both simulations and experiments featuring small mobile ground robots and aerial robots (drones), thus providing a proof of concept for robotic swarmalators. To the best of our knowledge, this is the first ''sync and swarm solution'' robustly working in robot systems and creating emerging space-time patterns.

The article is structured as follows: Section II covers related work. Section III introduces the discrete sync and swarm model. Section IV shows the obtained spatio-temporal patterns with order parameters used to distinguish them. Sections V and VI present a simulation-based analysis and an experimental proof of concept. Section VII concludes.

II. RELATED WORK

There is extensive literature on coordination in multi-agent systems across many scientific disciplines. We focus here on applied work in mobile robotics and wireless networks related to self-organizing synchronization and swarming.

The use of pulse-coupled oscillator synchronization in networked systems has been successfully demonstrated by a number of research groups. Perez-Diaz *et al.* [6] showed by experiments how the field of view and speed of movement influences synchronization of mobile robots. Brandner *et al.* [7] improved the precision of synchronization in wireless networks by equalizing the oscillation frequencies of the agents, where the algorithm was tested experimentally with programmable radios. Trianni and Nolfi [8] used evolutionary mechanisms to achieve synchronization in robot swarms. Although the robots aim at performing a synchronized movement, they do not interact in space, but only adjust their behavior to the oscillations. The aforementioned implementations of temporal coordination mechanisms differ in their communication interface. They use light [6], sound [8], and radio [7]. All these works, in contrast to this publication, focus only on the temporal coordination (phase interaction) between robots and do not consider their spatial influence.

Some researchers exploit synchronization to control swarm behavior. However, such approaches considered only one-directional coupling so far. Hartbauer and Römer [9] proposed a swarm of synchronized agents that uses the emitted pulses for navigation. Robots close to the goal increase their oscillation frequency so that the other agents follow their signals. Christensen *et al.* [10] proposed a method for detecting faulty agents in a swarm. The robots periodically emit light. If the light on any agent is not detected in time, that agent is considered to be broken and its task needs to be taken over. A method proposed by Bezzo *et al.* [11] uses synchronization to detect changes in the network topology, which was shown to maintain the formation of robots.

In terms of spatial coordination, multi-robot researchers are interested in different aspects of self-organization. Examples include swarm aggregation [12], flocking [13], and pattern formation [14]. A widespread technique is to use the theory of potential fields. One of the reasons why this approach became popular is that it can be used to fulfill different tasks, like navigation [15], formation control [16], and swarm aggregation, the last being most relevant to this work. Gazi [17] provided a formulation of the artificial potential for swarm aggregation and a controller that considers the dynamics of agents. The stability of swarm aggregation based on potential fields was analyzed by Fetecau *et al.* [18] and Gazi and Passino [19]. Tanner *et al.* [20] proposed conditions that need to be fulfilled by the potential to guarantee stable flocking. In most articles, the potential and its gradient need to be updated continuously or at least at a high rate. In this work, we aim at reducing the update rate, which, however, might in turn destabilize the swarm. Therefore, we use work by Armijo [21] to determine the maximum safe speed of robots.

The mechanisms of synchronization of coupled oscillators can be applied to stabilize collective motion of multiple robots [22]. Motivated by the application for underwater vehicles, Sepulchre *et al.* [23] presented a method based on the well-known Kuramoto model [24] that allows stabilization of parallel and circular motion of self-propelled particles. Gao and Wang [25] achieved similar results: stabilization of parallel and circular motion but based on pulse coupling. Depending on the desired behavior, they modify the phase response of the agents. This solution is easier to achieve if communication is restricted to discrete instants of time.

Although research on synchronization and swarming was somehow connected for some time, the mathematical model by O'Keeffe *et al.* [3] was presumably the first one taking into account the *mutual* coupling of these two phenomena. Their simulation results showed that agents using this unified model — the ''swarmalators'' — can form five spatio-temporal patterns, whose stability was analyzed in [26]. The model assumes continuous, delay-free coupling, which is impossible to achieve in multi-robot systems. Potential applications of the swarmalator model were presented by O'Keeffe and Bettstetter [27]. Our preparatory work [4] for this article was, as far as we know, the first to implement and showcase a swarmalator model in an engineered system.

III. MODEL

Each sandsbot is modeled as an agent $k \in \{1, \ldots, N\}$. It has an oscillator, whose value is represented by a phase $\Phi_k(t) \in [0, 2\pi)$ evolving over time t. The time index t is skipped if not needed. The phase difference between k and another agent j with phase Φ_j is denoted by Φ_{jk}. The spatial position of an agent is $\mathbf{x}_k \in \mathbb{R}^2$. The position difference vector from k to j located at $\mathbf{x}_j$ is $\mathbf{x}_{jk} = \mathbf{x}_j - \mathbf{x}_k$ and the Euclidean distance between them is $\|\mathbf{x}_{jk}\|$.

The system dynamics can be outlined as follows. The phases of different agents influence each other based on certain rules (temporal coordination). For example, they may

synchronize to a common value or "desynchronize" to differing ones. The positions of the agents influence each other as well (spatial coordination). For example, agents may physically attract or repel each other based on the distance between them. Most important for this work, the phases influence the movement, and the positions influence the phase dynamics (bidirectional coupling of temporal and spatial coordination). For example, agents with similar phases may attract or repel each other stronger, and close-by nodes may synchronize faster. This leads to the emergence of a space-time pattern, whose shape depends on the coupling parameters.

We describe how the system works with time-discrete interactions and how time delays are incorporated into the model design (Section III-A). Next, we show how the temporal coordination model controls the phases (Section III-B) and how the spatial aggregation model controls the movement dynamics (Section III-C). Finally, we present how the temporal and spatial coordination are coupled to create a unified model (Section III-D).[1]

A. TIME-DISCRETE INTERACTIONS

Each agent must repeatedly share its state (phase and position) with other agents. In real-world systems, these state updates can occur only at discrete points in time, and each message experiences processing and propagation delays. Moreover, the fact that the phase Φ_k is constantly changing and, if the agent is moving, the position $\mathbf{x}_k$ is changing, a simple sharing of the state would lead to the situation that the up-to-date state is unknown to the receiver at the time of reception. To tackle these issues, we split the phase into two components: a discrete phase θ_k and an oscillatory part ϕ_k, formally $\Phi_k = \phi_k + \theta_k$. Each of them plays an important role to enable robust interactions over a wireless medium.

The model of phase evolution is presented in Figure 1. The discrete phase θ_k is exchanged between agents rather than the overall continuous phase Φ_k. The value of θ_k remains constant throughout a period T, so the message can be sent and received by other agents before the value changes. To operate on integer-valued phases, we introduce the *phase level* $\hat{\theta}_k \in \{0, 1, \ldots, L-1\}$ with L being the number of phase levels. The discrete phase is then defined as $\theta_k = \frac{\hat{\theta}_k}{L} 2\pi$.

To maintain consistency, sandsbots operate in synchrony. For this purpose, we use the oscillatory part $\phi_k \in [0, \frac{2\pi}{L})$ as the internal clock, which oscillates with period T and is synchronized throughout the whole system. Such synchronization can be achieved by established techniques, where our implementation uses the Network Time Protocol (NTP). Each oscillation cycle is split into three parts:

- Whenever $\phi_k = 0$, the agent updates its state, computes the *predicted* state (phase level and predicted position) for the end of the oscillation cycle, and sends a message containing this information.

[1]The unpublished contributions of this section are mainly in Sections III-C and III-D and aspects related to spatial coordination in Section III-A. Section III-B and parts of Section III-A are taken, in modified form, from [5], to make this article self-contained and present the overall solution.

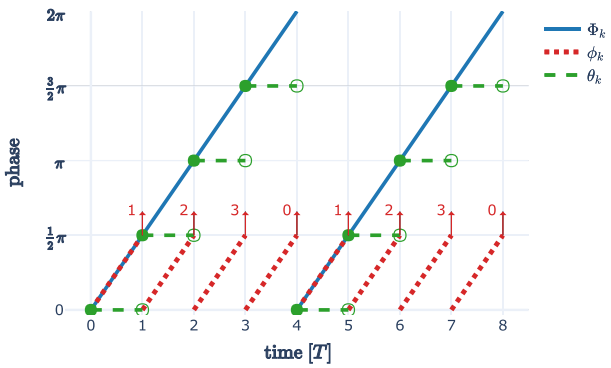


FIGURE 1. Phase of an agent Φ_k with its components θ_k and ϕ_k over normalized time. Arrows show the moment a signal is emitted; the numbers next to them represent the phase level. Figure taken from [5].

- For ϕ_k from 0 to a threshold $\phi_U \in [0, 2\pi/L)$, the agent gathers messages containing states of the other ones.
- For ϕ_k from ϕ_U to the maximum value $2\pi/L$, the agent calculates the new update of its state based on the received predicted states and its own predicted state.

The threshold ϕ_U can be chosen depending on the expected delays and the computational effort of calculating a new update (and thus the time needed). High values ensure that even delayed messages are taken into account but leaves less time to calculate the update; low values leave more time for the calculations with the risk of missing some delayed messages. The predicted position is calculated based on the current position, period length, and calculated velocity.

An agent can change its phase immediately but its position only slowly by setting a new velocity. Thus, the output of the sandsbot model (state update) consists of a new phase level and velocity.

Our model assumes that messages suffer from delays and takes these delays into account. If a message arrives before ϕ_k reaches the threshold ϕ_U, the agent uses the up-to-date data to calculate the next update (as messages contain a prediction, not the past state). Messages that arrive after reaching the threshold are dropped. This means that delays shorter than $T\phi_U/\frac{2\pi}{L}$ do not influence the performance. In our experiments, the period length T and threshold ϕ_U are chosen based on the measured latency in the network to ensure that the vast majority of messages will be delivered before the threshold ϕ_U is reached.

B. TEMPORAL COORDINATION

1) TEMPORAL PATTERNS

The sandsbots are required to arrange in three temporal patterns: *synchronized* where all agents have the same phase, *splay* where the phases are evenly spaced, and *clustered* where phases are arranged in evenly-spaced groups of equal size. The model proposed in our work [5] is used to achieve these patterns. Although phase levels differ in some patterns, internal clocks are not influenced and remain synchronized.

If M denotes the number of clusters, synchrony is achieved for $M = 1$, the splay pattern for $M = N$, and clusters for $M \in \{2, \ldots, N-1\}$, where only clusters of equal size are considered, i.e., $M \mid N$ (M divides N). The number of phase levels should be chosen such that $M \mid L$, which allows to maintain equal distances between clusters. These patterns are called *M-clusters*. All three desired patterns are M-clusters with synchrony and splay being the extreme cases.

The phase θ_k of an agent k can be written as a complex number $e^{\mathrm{i}\theta_k}$. A freely running oscillator is thus modeled as rotating vector of unit length. In a system of N agents, the m-th moment of the N phases is [23]

$$z_m = \frac{1}{Nm} \sum_{k=1}^{N} e^{\mathrm{i}m\theta_k}. \tag{1}$$

The term $m\theta_k$ can be interpreted as the m-th harmonic of θ_k, and (1) is sometimes called complex-valued order parameter [24] of the m-th harmonic [28].

The magnitude $r_m = |z_m|$ reveals the state in which the system is: The phases of the m-th harmonic are synchronized if $r_m = \frac{1}{m}$ and balanced if $r_m = 0$. To achieve an M-cluster, all phases of the first $M-1$ harmonics are balanced ($\forall m \in \{1, \ldots, M-1\}$. $r_m = 0$) and the phases of the M-th harmonic are synchronized ($r_M = \frac{1}{M}$) [23]. To give some examples, synchronized agents have $r_1 = 1$, a two-cluster pattern yields $r_1 = 0$ and $r_2 = \frac{1}{2}$, and a three-agent splay pattern has $r_1 = r_2 = 0$ and $r_3 = \frac{1}{3}$.

2) PHASE POTENTIAL AND COUPLING FUNCTIONS

Based on the order parameter, we define the potential $U_m = \frac{N}{2} K_m r_m^2$, similar as in [23], where K_m is the coupling strength of the m-th harmonic. If $K_m < 0$ the potential reaches its minimum if the phases of the m-th harmonic are synchronized. In contrary, if $K_m > 0$ the phases of the m-th harmonic need to be balanced for U_m to be minimal. To have the first $M-1$ harmonics balanced and the M-th harmonic synchronized, we set $K_m > 0$ for $m \in \{1, \ldots, M-1\}$ and $K_M < 0$. The overall potential of an M-cluster is the sum $U^{(M)} = \sum_{m=1}^{M} U_m$ [23], which reaches its global minimum if each U_m is minimized. For the given coupling strengths, this minimum corresponds to the M-cluster pattern [23].

In a system of agents with continuous phases, the overall potential can be minimized with gradient control [23]:

$$\dot{\Phi}_k = \omega_k - \frac{1}{N} \frac{\partial U^{(M)}}{\partial \Phi_k}, \tag{2}$$

where ω_k is the agent's natural frequency of oscillations. The derivative can be defined as the sum of phase coupling functions $\Gamma(\Phi_{jk})$ between the agents [23]:

$$\frac{\partial U^{(M)}}{\partial \Phi_k} = \sum_{j=1}^{N} \Gamma(\Phi_{jk}). \tag{3}$$

This results in a phase coupling function being a linear combination of the continuous couplings, similar to the Kuramoto model [24], for the first M phase harmonics [23]:

$$\Gamma(\Phi_{jk}) = \sum_{m=1}^{M} \frac{K_m}{m} \sin(m\Phi_{jk}). \tag{4}$$

3) FROM CONTINUOUS TO DISCRETE PHASE COUPLING

The model presented until now is valid for continuous phase and continuous time. For discrete phase θ_k and discrete time, we propose a *phase interaction function*:

$$\Psi_k(\theta_k) = \frac{1}{N} \sum_{j=1}^{N} \Gamma(\theta_{jk}), \tag{5}$$

which uses the same phase coupling function $\Gamma(\cdot)$ but now evaluated for discrete phase values.

We split the phase coupling function into two additive parts based on the value of K_m:

$$\Gamma(\theta_{jk}) = \Gamma_1(\theta_{jk}) + \Gamma_2(\theta_{jk}), \tag{6}$$

$$\Gamma_1(\theta_{jk}) = \sum_{m \in \mathscr{M}_1} \frac{K_m}{m} \sin(m\theta_{jk}), \tag{7}$$

$$\Gamma_2(\theta_{jk}) = \sum_{m \in \mathscr{M}_2} \frac{K_m}{m} \sin(m\theta_{jk}), \tag{8}$$

with $\mathscr{M}_1 = \{m \mid K_m \leq 0\}$ and $\mathscr{M}_2 = \{m \mid K_m > 0\}$, which corresponds to $\mathscr{M}_1 = \{M\}$ and $\mathscr{M}_2 = \{m \mid 1 \leq m < M\}$ for the M-cluster pattern. Γ_1 can be thought of as the *phase attraction function* that tries to synchronize harmonics belonging to $\mathscr{M}_1$ whereas Γ_2 is the *phase repulsion function* that balances the other phase harmonics. The split into attraction and repulsion gives the phase coupling a similar structure as the position coupling used for spatial coordination (14) and enables us to combine the two models (Section III-D).

The discrete temporal coordination operates on the integer phase levels. The phase interactions are usually too minor for an agent to immediately jump to the next phase level, especially if the system is close to reaching the desired pattern. Thus, each sandsbot integrates these interactions over time and keeps them as *phase level correction* $\delta\hat{\theta}_k$. Only if this value is large enough to change the phase by a full level, the phase level $\hat{\theta}_k$ is adjusted and the correction $\delta\hat{\theta}_k$ is reset.

There are multiple equilibrium states in such a dynamic system. For example, the synchronized pattern is also an equilibrium for the splay pattern because the agents belonging to a certain phase cluster do not influence each other ($\sin(\theta_{jk}) = 0$) and each of them is equally influenced by the agents outside the cluster. Therefore, a cluster formed once will never break. The situation is different in a system with continuous phases, where some noise is sufficient to disturb and eventually break a cluster. In contrast, a system with discrete phase is meant to prevent the influence of disruptions (e.g., communication delays, oscillators imperfections, noise). To break such unwanted clusters, we update the correction of the phase level $\hat{\theta}_k$ at time t as follows:

$$\delta\hat{\theta}_k[t] = (1+E)\delta\hat{\theta}_k[t-1] + \eta - \Psi_k(\theta_k[t]) \tag{9}$$

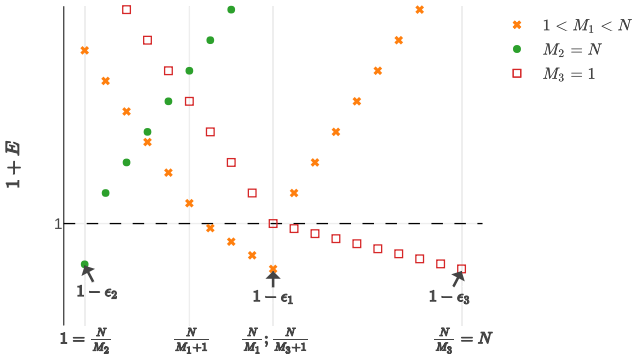


FIGURE 2. Shape of energy function for different sizes of clusters. The term $-\epsilon_k$ denotes minimal energy. Figure taken from [5].

with *noise* η and *state energy* E. The energy also stabilizes the desired M-cluster pattern. To achieve both breaking and stabilization, we need positive E for agents in a cluster that is either too big (more than $\frac{N}{M}$ agents) or much too small (at most $\frac{N}{2M}$ agents) and negative E otherwise (for agents being in a cluster of the target or almost target size). The shape of energy function is given in Figure 2 for different cluster sizes.

Whenever the internal clock resets ($\phi_k = 0$) the phase is updated. The phase level $\hat{\theta}$ is always incremented by 1 and the phase correction rounded towards 0 is applied:

$$\hat{\theta}_k[t+1] = \left(\hat{\theta}_k[t] + 1 + \operatorname{sgn}(\delta\hat{\theta}_k[t]) \left\lfloor \left|\delta\hat{\theta}_k[t]\right| \right\rfloor\right) \bmod L\,. \tag{10}$$

If the rounded phase correction was non-zero, it is reset ($\delta\hat{\theta}_k[t] = 0$).

The presented temporal coordination model allows agents to form synchronized, splay, and cluster patterns.

C. TIME-DISCRETE SPATIAL AGGREGATION

1) AGGREGATION, COLLISION AVOIDANCE, AND DEMANDED VELOCITY

The locations of the agents and their dynamics is controlled by an aggregation model based on potentials. The potential created by each agent represents two counteracting forces: attraction and repulsion. Attraction helps agents to gather; repulsion preserves spacing to avoid collisions.

To guarantee collision-free operation, we use a *safety area* around each agent. This area comprises a circular *bounding box* with diameter d and an additional *safety margin* ϵ_d defining the minimum distance that needs to be kept between the bounding boxes of agents at all times. The safety area of agent k is thus a circle of diameter $d + \epsilon_d$ centered at $\mathbf{x}_k$.

The potential at position $\mathbf{x}$ generated by the agent j is (similar to [18]):

$$\mathscr{V}_j(\mathbf{x}) = \eta_{\text{attr}} \frac{\left\|\mathbf{x}_j - \mathbf{x}\right\|^2}{2} - \eta_{\text{rep}} \ln\left(\left\|\mathbf{x}_j - \mathbf{x}\right\| - d\right), \tag{11}$$

for non-overlapping bounding boxes (i.e., $\left\|\mathbf{x}_j - \mathbf{x}\right\| - d > 0$), and undefined if the bounding boxes overlap. The scaling factors η_{attr} and η_{rep} adjust the pattern size. In the repulsion part of the potential, we take into account the distance between bounding boxes to assure that the repulsion goes to infinity if agents are close to collision (bounding boxes are almost tangent). The potential of agent k can be described as the average of the potentials created by other agents at this time:

$$V_k = \frac{1}{N} \sum_{\substack{j=1 \\ j\neq k}}^{N} \mathscr{V}_j(\mathbf{x}_k). \tag{12}$$

Similar to temporal coordination, each agent aims at minimizing its potential. In order to find this minimum we use the gradient descent method. The demanded velocity of agent k is defined as $\mathbf{v}_k^{\text{d}} = \nabla V_k$, which yields for (12):

$$\mathbf{v}_k^{\text{d}} = \frac{1}{N} \sum_{\substack{j=1 \\ j\neq k}}^{N} \nabla \mathscr{V}_j(\mathbf{x}_k)\,, \tag{13}$$

where the summands can be expressed as

$$\nabla \mathscr{V}_j(\mathbf{x}_k) = \mathbf{I}_1(\mathbf{x}_{jk}) - \mathbf{I}_2(\mathbf{x}_{jk}) \quad \text{with} \tag{14}$$

$$\mathbf{I}_1(\mathbf{x}_{jk}) = \eta_{\text{attr}}\, \mathbf{x}_{jk}\,, \tag{15}$$

$$\mathbf{I}_2(\mathbf{x}_{jk}) = \frac{\eta_{\text{rep}}}{\left\|\mathbf{x}_{jk}\right\| \left(\left\|\mathbf{x}_{jk}\right\| - d\right)} \mathbf{x}_{jk} \tag{16}$$

describing how strong agent k is attracted or repelled by the other agents.

2) CONSIDERATION OF ROBOT CONSTRAINTS

The maximum speed $v_k^{\max}$ depends on physical and safety constraints. The major physical constraint is the maximum possible speed v_k^{maxR} of the robot platform in use. The safety constraint can be formulated as

$$v_k^{\text{maxS}} = \frac{\min_{j\neq k}\left(\left\|\mathbf{x}_{jk}\right\|\right) - d - \epsilon_d}{4T}. \tag{17}$$

This constraint ensures that during one period each agent can move at most a quarter of the gap between its own safety area and the one of the closest neighbor. The distance between safety areas of each pair of agents will change by not more than half, which should not only guarantee that the agents never collide but also that their bounding boxes do not come closer than ϵ_d. When agents get very close to each other, the repulsion between them is very high. It can be especially risky if one agent is surrounded by a few other ones. The safety constraint additionally slows down agents operating in a close proximity, thus preventing rapid reactions if two agents get close to each other. As a result of the constraints, the maximum speed is $v_k^{\max} = \min(v_k^{\text{maxR}}, v_k^{\text{maxS}})$. Based on this, the velocity of agent k is

$$\mathbf{v}_k = \min\left(\left\|\mathbf{v}_k^{\text{d}}\right\|, v_k^{\max}\right) \cdot \frac{\mathbf{v}_k^{\text{d}}}{\left\|\mathbf{v}_k^{\text{d}}\right\|}\,, \tag{18}$$

and the position change during one oscillation cycle is

$$\mathbf{x}_k[t] = \mathbf{x}_k[t-1] + T\, \mathbf{v}_k[t-1]. \tag{19}$$

3) MITIGATION OF PHYSICAL OSCILLATIONS

The simulation of this model shows an unwanted behavior: sandsbots oscillate around their positions. To compensate this phenomenon, we modify the maximum speed based on a theorem introduced by Armijo [21], which states: If the gradient of function f is Lipschitz continuous — i.e., $\forall x, y \in \mathscr{D}_f$: $\|\nabla f(x) - \nabla f(y)\| < \lambda \|x - y\|$, where $\mathscr{D}_f$ is a domain of f — gradient descent converges for step size $s \leq \frac{1}{2\lambda}$. Hence, to avoid spatial oscillations, we introduce an additional term $s\left\|\mathbf{v}_k^{\mathrm{d}}\right\|$ in the velocity equation (18) to limit the speed of each agent:

$$\mathbf{v}_k = \min\left(\left\|\mathbf{v}_k^{\mathrm{d}}\right\|, s\left\|\mathbf{v}_k^{\mathrm{d}}\right\|, v_k^{\max}\right) \cdot \frac{\mathbf{v}_k^{\mathrm{d}}}{\left\|\mathbf{v}_k^{\mathrm{d}}\right\|}, \tag{20}$$

where

$$s = \frac{1}{2\lambda T} \tag{21}$$

is a step size calculated based on period length and the theorem. This modification limits the speed to guarantee that the step size is short enough for the gradient descent to converge.

By limiting the speed due to the safety constraint (17), we guarantee that the minimum distance constraint

$$\forall k, j \quad \left\|\mathbf{x}_{jk}\right\| > d + \epsilon_d \tag{22}$$

is met. Hence, we can limit the domain of V_k, so that $\left\|\nabla^2 V_k\right\|$ is bounded above and the gradient is Lipschitz continuous with the Lipschitz constant $\lambda = \eta_{\mathrm{attr}} - \eta_{\mathrm{rep}}/\epsilon_d^2$.

This approach enables us to slow down the agent if the demanded velocity would lead to oscillations. To achieve faster convergence, the Lipschitz constant can be computed for the gradient limited only to the neighborhood of the current position, with the assumption that the agents can move in any direction at its maximum speed.

D. COUPLING OF TEMPORAL AND SPATIAL COORDINATION

We extend the above models to propose a time-discrete solution in which temporal and spatial coordination are mutually coupled. The overall result is given in Box 1 used with the interaction functions specified in Box 2.

1) INFLUENCE OF PHASE ON SPATIAL COORDINATION

First we introduce how phases influence the position interactions. The demanded velocity of this phase-influenced aggregation model has the following form (similar to [26]):

$$\mathbf{v}_k^{\mathrm{d}} = \frac{1}{N}\sum_{\substack{j=1\\ j\neq k}}^{N} \mathbf{I}_1(\mathbf{x}_{jk})F_1(\theta_{jk}) - \mathbf{I}_2(\mathbf{x}_{jk})F_2(\theta_{jk}), \tag{23}$$

where functions $F_1(\theta_{jk}) = 1 + J_1\cos(\theta_{jk})$ and $F_2(\theta_{jk}) = 1 - J_2\cos(\theta_{jk})$ describe how the agents' phase similarity influences their spatial attraction and repulsion, respectively, with parameters J_1 and J_2 defining the strength of these influences. The attraction of agents with similar phases is stronger than in the aggregation model if $J_1 > 0$, it remains unchanged for $J_1 = 0$, and it is weaker if $J_1 < 0$. We set $J_2 = 0$ to prevent collisions. Using a high J_2 value could cause agents with similar phases not to repel or even attract each other, no matter how close they are, which could lead to collisions.

Since the phases now influence the positions, the previously calculated step size (21) does not guarantee oscillation-free convergence. The reason being that, once a phase changes, attraction and thus velocity might change significantly. There are two options to compensate for this behavior: using more phase levels ($L \gg M$) or keeping the same number of phase levels and changing the step size. We use the step size

$$s = \frac{1}{2\lambda T} \cdot G\big(\hat{U}^{(M)}\big), \tag{24}$$

where $\hat{U}^{(M)} \in [0, 1]$ is a normalized value of the potential, which is 1 if $U^{(M)}$ is maximal and converges to 0 when agents' phases reach the desired pattern. The function $G(\cdot)$ defines how much phase potential should influence the step size of the spatial aggregation. In this publication we use:

$$G\big(\hat{U}^{(M)}\big) = 1 - \sqrt[p]{\hat{U}^{(M)}}, \tag{25}$$

which allows us to control which model has priority. High values of p will make agents move slowly when their phases have not formed the correct pattern yet. Low values will make agents move dynamically even before the temporal pattern has converged. We use $p = M$, which means: the more clusters should be formed, the less dynamic the movements are before the temporal pattern has converged.

2) INFLUENCE OF POSITION ON TEMPORAL COORDINATION

The phase interactions are modified in a similar way. We enable the distance between agents to influence coupling of each phase harmonic:

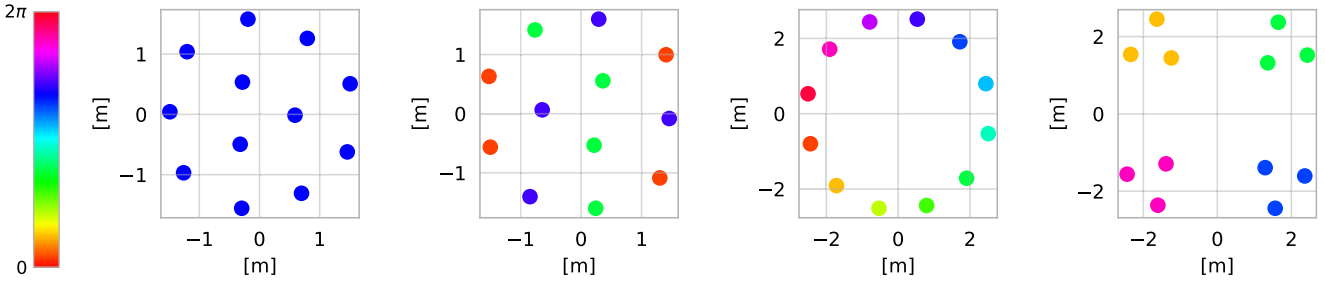
$$\Psi_k(\theta_k) = \frac{1}{N}\sum_{\substack{j=1\\ j\neq k}}^{N} \Gamma_1(\theta_{jk})\Lambda_1(\mathbf{x}_{jk}) + \Gamma_2(\theta_{jk})\Lambda_2(\mathbf{x}_{jk}), \tag{26}$$

where the functions Λ_1 and Λ_2 define how the distance between agents influences their phase attraction and repulsion, respectively. We use the following form of these functions:

$$\Lambda_1(\mathbf{x}_{jk}) = 1 + P\frac{\epsilon_d + d}{\left\|\mathbf{x}_{jk}\right\|}; \tag{27}$$

$$\Lambda_2(\mathbf{x}_{jk}) = 1 - P\frac{\epsilon_d + d}{\left\|\mathbf{x}_{jk}\right\|}. \tag{28}$$

The parameter $P \in [-1, 1]$ quantifies the strength of influence of distance between agents on their phase coupling. For positive P, when agents move closer to each other, their phase attraction is amplified and their phase repulsion is weakened. If $P = 0$, the positions do not influence the



(a) Static sync: $J_1 = 0.1$, $K = 1$, $M = 1$, $P = 0.1$.

(b) Static async: $J_1 = -1$, $K = 0.2$, $M = 3$, $P = -0.03$.

(c) Static phase wave: $J_1 = 1.4$, $K = 1$, $M = N = 12$, $P = 0.1$.

(d) Splintered phase wave: $J_1 = 1.5$, $K = 1$, $M = 4$, $P = 0.1$.

FIGURE 3. Spatio-temporal patterns and parameter values for which they were obtained. The color map used for visualization of phases is shown on the left.

phases. The influence of spatial proximity is strongest at the shortest distance between agents ($\epsilon_d + d$) and decreases with increasing distance.

Box 1: Sandsbots model

$$\mathbf{v}_k^{\mathrm{d}} = \frac{1}{N}\sum_{\substack{j=1\\ j\neq k}}^{N} \mathbf{I}_1(\mathbf{x}_{jk})F_1(\theta_{jk}) - \mathbf{I}_2(\mathbf{x}_{jk})F_2(\theta_{jk})$$

$$\mathbf{v}_k = \frac{\mathbf{v}_k^{\mathrm{d}}}{\left\|\mathbf{v}_k^{\mathrm{d}}\right\|} \cdot \min\left(\left\|\mathbf{v}_k^{\mathrm{d}}\right\|, s\left\|\mathbf{v}_k^{\mathrm{d}}\right\|, v_k^{\max}\right)$$

$$\mathbf{x}_k[t+1] = \mathbf{x}_k[t] + \min(s, T)\mathbf{v}_k$$

$$\Psi_k(\theta_k) = \frac{1}{N}\sum_{\substack{j=1\\ j\neq k}}^{N} \Gamma_1(\theta_{jk})\Lambda_1(\mathbf{x}_{jk}) + \Gamma_2(\theta_{jk})\Lambda_2(\mathbf{x}_{jk})$$

$$\delta\hat{\theta}_k[t] = (1+E)\delta\hat{\theta}_k[t-1] + \eta - \Psi_k(\theta_k[t])$$

$$\hat{\theta}_k[t+1] = \left(\hat{\theta}_k[t] + 1 + \mathrm{sgn}(\delta\hat{\theta}_k[t]) \cdot \left\lfloor\left|\delta\hat{\theta}_k[t]\right|\right\rfloor\right) \bmod L$$

$$\delta\hat{\theta}_k[t] = 0 \text{ if phase was corrected}$$

Box 2: Interaction functions

$$\mathbf{I}_1(\mathbf{x}_{jk}) = \eta_{\mathrm{attr}}\mathbf{x}_{jk}$$

$$\mathbf{I}_2(\mathbf{x}_{jk}) = \eta_{\mathrm{rep}}\frac{\mathbf{x}_{jk}}{\left\|\mathbf{x}_{jk}\right\|\left(\left\|\mathbf{x}_{jk}\right\| - d\right)}$$

$$F_1(\mathbf{x}_{jk}) = 1 + J_1\cos(\theta_{jk})$$

$$F_2(\mathbf{x}_{jk}) = 1 - J_2\cos(\theta_{jk})$$

$$\Gamma_1(\theta_{jk}) = \sum_{m\in\mathscr{M}_1}\frac{K_m}{m}\sin(m\theta_{jk})$$

$$\Gamma_2(\theta_{jk}) = \sum_{m\in\mathscr{M}_2}\frac{K_m}{m}\sin(m\theta_{jk})$$

$$\Lambda_1(\mathbf{x}_{jk}) = 1 + P\frac{\epsilon_d + d}{\left\|\mathbf{x}_{jk}\right\|}$$

$$\Lambda_2(\mathbf{x}_{jk}) = 1 - P\frac{\epsilon_d + d}{\left\|\mathbf{x}_{jk}\right\|}$$

IV. SPATIO-TEMPORAL PATTERNS

We now show that the sandsbot model is able to reproduce the patterns of the continuous sync and swarm model [3]. Some patterns are modified to guarantee they will emerge regardless of the initial conditions (static phase wave) or to control their properties (static async, splintered phase wave). Such an approach allows to adjust the pattern for a specific task that needs to be executed. We show the position and phase of each agent, where the phase is indicated by a color. The color map is presented on the left side of Figure 3.

A. PATTERNS

In the *static sync* pattern (Figure 3a), all agents synchronize their phases and gather evenly distributed on a disk. This state is formed if $J_1 > -1$, $K > 0$, $M = 1$, and $P \geq 0$.

The *static async* pattern distributes the agents on a disk as well, but their phases are now asynchronous and similar phases spread in space. Using the parameters of the original model [3] for this state ($K < 0$, $J_1 < 0$) and keeping $M = 1$, the discrete phase levels make the obtained pattern unstable with phases constantly changing. Thus, we introduce a controlled version of this pattern (Figure 3b) in which we specify the number of clusters M to be created ($1 < M < N$) and set $K > 0$, $J_1 < 0$, and $P < 0$. Agents with different phases attract each other more and they try to form clusters, but phase interaction between agents is stronger if they are more distant from each other. Agents form M clusters in the phase domain and each phase cluster spreads on a disk.

The *static phase wave* (Figure 3c) formed by our model looks similar to the one in [3]. The agents form an annulus and sort their uniformly distributed phases. Sandsbots can form this pattern regardless of their initial conditions, contrary to the original model [3], in which phase coupling does not exist in this pattern. This pattern appears for $K > 0$, $J_1 > 0$, $M = N$, and $P > 0$. In a special case of this pattern (called ring phase wave in [26]) the agents are placed on a circle.

The *splintered phase wave* of [3] splits the agents in space into clusters with similar phases. The clusters are positioned on a ring. The number of clusters formed depends on the initial conditions, and the agents keep changing phases slightly and move within their clusters. Similarly to static async, we propose a controlled version, in which we specify the

number of clusters and set $K > 0, J_1 > 0$, and $P > 0$. Agents cluster in the phase domain, are placed on a ring, and split into clusters based on their phase. After reaching this pattern, the positions and phases remain unchanged.

Agents in the *active phase wave* form a ring and keep changing their phase while they travel around the ring to get close to the ones having a similar phase. This pattern is achieved for $K < 0, J_1 > 0, M = 1$, and $P > 0$. As we do not control the number of clusters directly here, $\hat{U}^{(M)}$ never converges to 0. Therefore, to speed up the formation, the influence of the phase potential on the step size s might be disabled ($G = 1$). Although we focus on stationary patterns, we reproduce this pattern for the sake of completeness, to show that it is possible to obtain similar behavior with discrete phase, but do not analyze it further.

B. ORDER PARAMETERS

Three parameters are used to characterize the different space-time patterns and to distinguish them formally.

1) SYNCHRONIZATION ORDER PARAMETER

Recall from Section III-B that the magnitude r_1 of the complex order parameter, simply denoted r in the following, is a measure of synchrony. It can be used to distinguish static sync ($r = 1$) from other patterns, as static phase wave (splay state in the phase domain), static async, and splintered phase wave (clustered states in the phase domain) are special cases of the balanced phase state ($r = 0$).

2) SWARMING ORDER PARAMETER

Three types of spatial arrangement occur: circle (ring phase wave), disk (static sync, static async), and annulus (splintered phase wave, static phase wave). To distinguish them formally, we propose to use the normalized variance of the distances d_k of the agents from their centroid, i.e.,

$$V = \frac{\sigma_d}{\sigma_{d'}(R)} = \frac{1}{N}\sum_{k=1}^{N}\frac{(d_k - \mu_d)^2}{\sigma_{d'}(R)}, \tag{29}$$

and call it *swarming order parameter* $V \in \mathbb{R}_0^+$. The term μ_d is the mean distance from the centroid. For normalization, we use the variance of the distance d' from the center of a disk with radius R to a point chosen uniformly at random from this disk, where we assume that the radius is $R = \max_k(d_k)$.

We observe the following: Agents placed on a circle yield $V = 0$, agents on an annulus have $0 < V < 1$, and agents distributed on a disk lead to $V > 1$. If the disk is fully occupied (i.e., the agents are tightly packed, minimum distances are not preserved), V is equal to 1, but in practice it is higher due to the minimum distance constraint (22).

3) CORRELATION BETWEEN ANGULAR POSITION AND PHASE

To express the correlation between the phases θ_k and the angular positions γ_k of the agents, we use the order parameter $S = \max(S_+, S_-)$ with [3]

$$S_\pm = \frac{1}{N}\left|\sum_{k=1}^{N} e^{\mathrm{i}(\gamma_k \pm \theta_k)}\right|, \tag{30}$$

which varies from 0 (no correlation) to 1 (perfect correlation). The angular position is $\gamma_k = \arctan(y_k/x_k)$ with the coordinates x_k and y_k with respect to the centroid of the whole swarm. Perfect correlation occurs in the static phase wave.

V. SIMULATION-BASED ANALYSIS

A. SETUP

The sandsbot model is analyzed in more detail with a simulation implemented in *Python*. We first study how to form patterns with the discrete model in perfect conditions. The imperfections of robots and issues associated with communication are not taken into account. Agents are connected without delays and packet loss. This leads to full knowledge about phase levels and positions of other agents and perfect synchronization of the clocks. The agents can move freely in space, the only constraint being their maximum speed. Real-world issues related to communication, movement constraints, and hardware imperfections are addressed later in our proof of concept with robotic platforms (Section VI).

All agents start with random phase levels and random positions in a 10 m × 10 m square both drawn from the uniform distribution. The initial positions are redrawn until they meet the minimum distance constraint (22). The clocks ϕ_k are synchronized and $\delta\hat{\theta}_k = 0$. Simulations are run with $T = 0.125$ s, $\epsilon_d = 0.1$ m, $d = 0.2$ m, and $v^{\mathrm{maxR}} = 0.2$ m/s, where this choice of values is motivated by capabilities of robots and aims at simulating a setup similar to the experimental one.

B. RESULTS

1) ORDER PARAMETERS

The plots in Figure 4 show for each of the static patterns how the three order parameters evolve over time. In each plot, three moments are marked (dashed black line) for which the corresponding space-time patterns are shown below. These moments are chosen to depict the starting condition, process of pattern formation, and the final pattern.

The plots confirm that for all patterns the order parameters converge to the values described in Section IV-B. This shows that the introduced combination of order parameters can serve as a tool to distinguish the patterns. The splintered phase wave (Figure 4d) forms very slowly. However, the state similar to the final pattern is formed much earlier. Around $t = 130$ s agents already form the correct temporal pattern and the clusters are placed on a ring. After that, the interactions between agents are minimal and the clusters slowly rotate to reach their final positions.

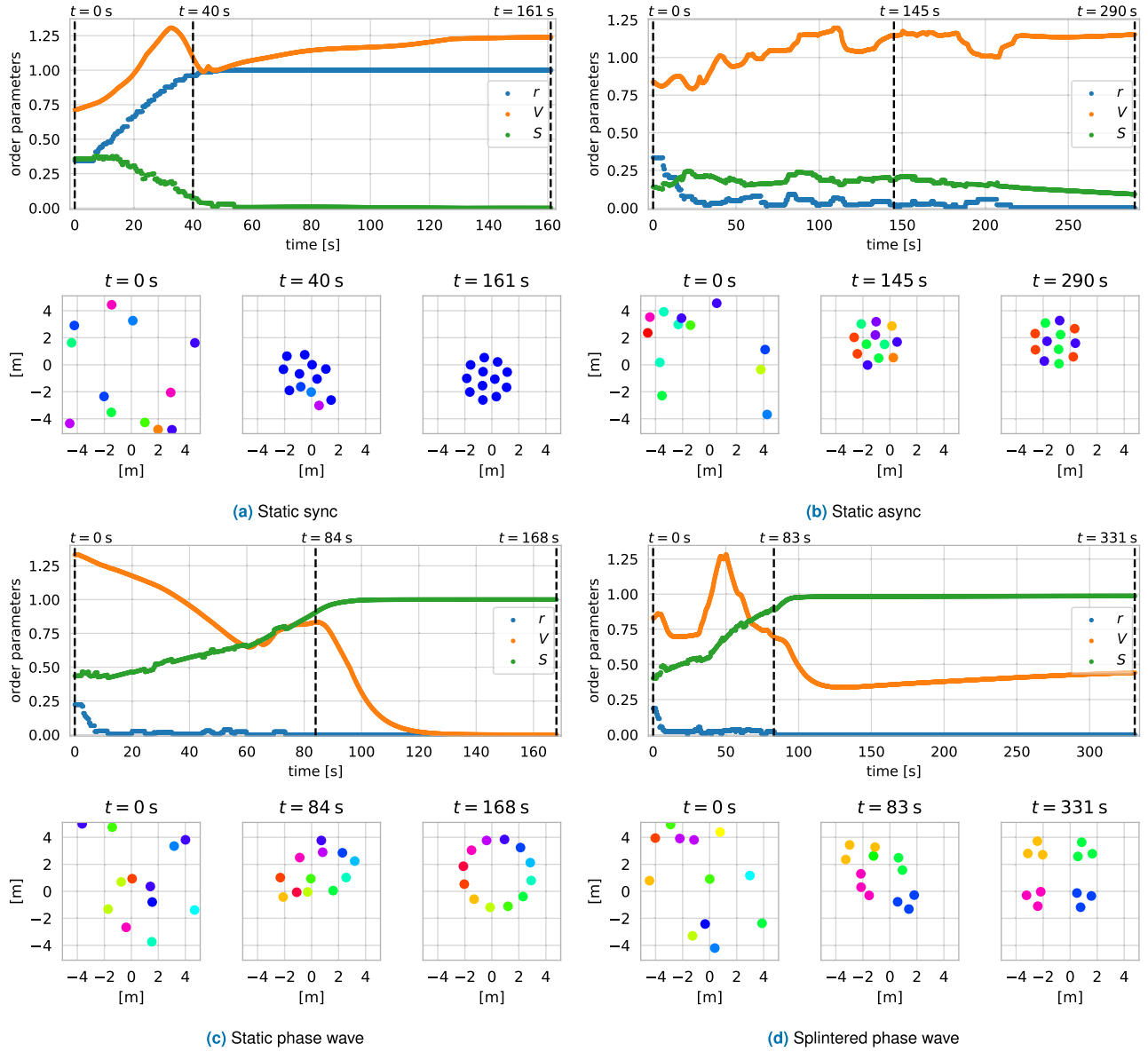


FIGURE 4. Convergence of order parameters and snapshots of the system showing the process of pattern forming.

2) IMPACT OF PERIOD LENGTH ON PATTERN CONVERGENCE

We now analyze the impact of the period length T and thus the frequency of message exchange on the convergence time and capabilities of the model. Sandsbots are compared to swarmalators with interaction functions from [26]. To ensure fair comparison, we add collision avoidance and maximum speed to the swarmalator model (as done in [4]).

A comparison is possible only for the static sync pattern as it is the sole pattern emerging identically in both models. The static async pattern and splintered phase wave with sandsbots differ significantly from their theoretical counterparts. The static phase wave now involves phase interactions and forms regardless of the initial conditions although it might converge more slowly.

We observed that sandsbots successfully form the static sync pattern even with very long periods ($T = 5$ s), but forming takes significantly longer compared to short periods ($T = 0.1$ s). In contrary, with swarmalators (originally continuous in nature), increasing the period leads to destabilization and physical oscillations ($T = 1$ s) and eventually prevents forming the pattern all together ($T = 5$ s).

It is assumed that the pattern is formed when all agents move slower than 1 mm/s. The relationship between T and convergence time is presented in Figure 5. The swarmalator model with $T = 0.1$ s (as used in the simulations in [3])

serves as a baseline. It can be observed that for very short periods (in the order of 0.1 s) the convergence time is similar to the one obtained with the swarmalator model and grows for increasing period length. If the pattern needs to be formed fast but at the same time the network load should be minimized, the following technique can be used: start with a short period (in the order of 0.1 s) to form the pattern and then increase it to keep the pattern stable.

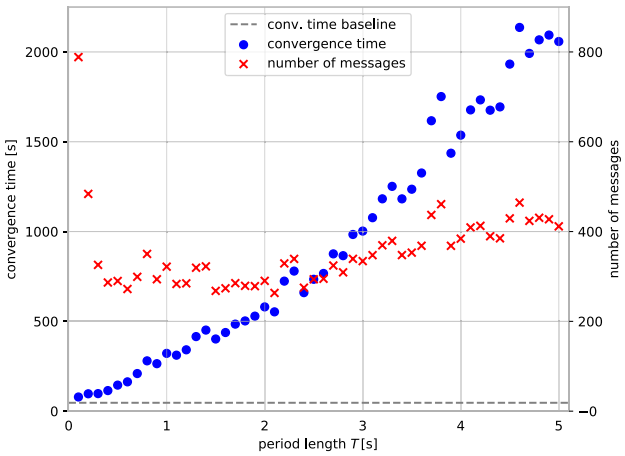


FIGURE 5. Relationship between period length and convergence time and the number of exchanged messages for static sync pattern.

Figure 5 also shows the number of exchanged messages for different period lengths. An interesting observation is that for very short periods (0.1 s and 0.2 s) the number of exchanged messages required to form the pattern is much higher than for longer periods. This happens because the robots could travel safely with their maximum speed for a time longer than the period, but they are forced to exchange data anyway. It means that the pattern formation for such short periods is inefficient in terms of the number of exchanged messages. For longer periods, the number of messages grows slowly with the period length due to the adaptive speed limit (20).

VI. EXPERIMENTAL VALIDATION

Finally, we validate the feasibility of the sandsbots model with two robot platforms. First, we use Crazyflie drones (Figure 6) to check whether the simple model of robot dynamics is sufficient and whether the discrete model works correctly even with imperfect estimation of future positions. Second, we use Pololu Balboa self-balancing robots (Figure 10) to see how the model behaves under realistic communication conditions with non-deterministic delays and message drops. We describe the setup for each platform and the results achieved in practical experiments. In each experiment, robots start from arbitrary positions with random phases.

A. CRAZYFLIES

Eight Crazyflies are controlled by a server, which acquires the positions of all from an Optitrack motion capture system.

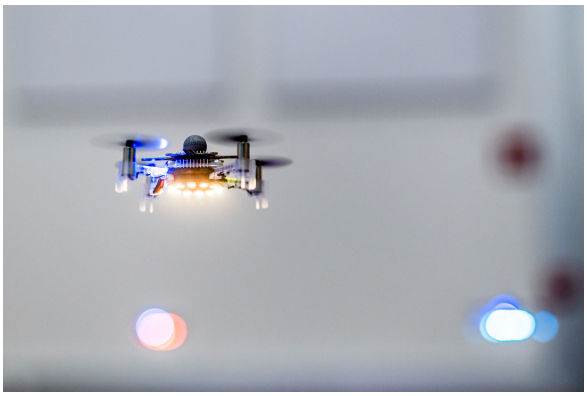


FIGURE 6. Crazyflie.

FIGURE 7. *Static sync*: Crazyflies forming the pattern.

This guarantees that full information is available without delay to calculate state updates for each agent, thus allowing us to omit potential communication problems. At the same time, the movement dynamics of the agents is realistic. At the beginning of each oscillation cycle (i.e., whenever $\phi_k = 0$) the server transmits new velocity and color to visualize the phase on a ring of RGBW light-emitting diodes (LEDs) attached to each robot. This experiment is run with the following parameters: $T = 0.5\,\mathrm{s}$, $\epsilon_d = 0.1\,\mathrm{m}$, $d = 0.3\,\mathrm{m}$ and $v^{\mathrm{maxR}} = 0.2\ \mathrm{m/s}$. Because of a constrained communication interface we use $\omega = \frac{2}{L}\mathrm{s}^{-1}$, which means that the velocity and color are updated twice per second.

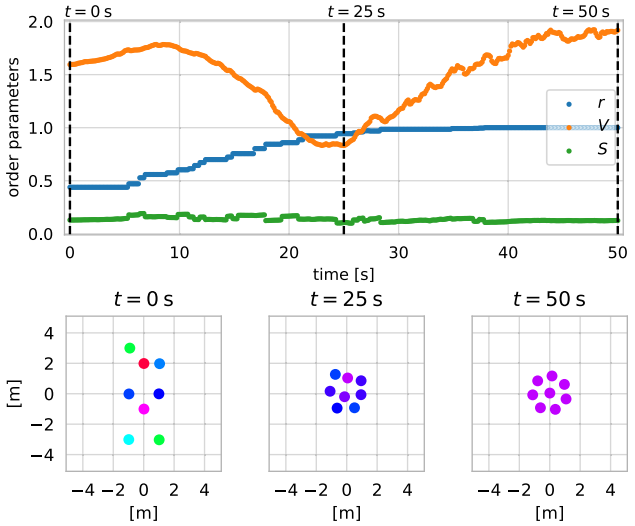


FIGURE 8. ***Static sync*: Convergence of order parameters and snapshots of the system showing the process of pattern forming with Crazyflies.**

All patterns are successfully formed by the Crazyflies. Figure 7 shows an exemplary pattern. Similar to the results of simulations, we studied how the order parameters converge for this pattern (Figure 8).

B. BALBOAS

For further evaluation — taking into account both movement dynamics and realistic communication — we use a robot swarm platform based on Pololu Balboa robots. The main computer is a Raspberry Pi 3B+. Robots communicate via IEEE 802.11bg operating in ad-hoc mode, which enables them to join and leave without any infrastructure or single point of failure. From the software perspective, the communication is realized in the ROS 2 framework (Eloquent Elusor release) using the Data Distribution Service (DDS) communication standard, which applies a Real-Time Publish Subscribe (RTPS) protocol (we use eProsima Fast RTPS). The robots communicate in best-effort mode with multicast enabled to reduce communication load.

The robots need to know their positions. Outdoors they could use a satellite-based positioning system, but as our demonstrator operates indoors, we utilize an Optitrack motion capture system. To ensure that the demonstrator will be easily transferable to real-world applications (including outdoors), the robots use the motion capture system in the same way as they would use an outdoor solution: each robot acquires only its own position and ignores messages sent to other agents. Each robot visualizes its phase level by the hue of the color of an LED strip attached to the bumpers.

The wireless channel is only used to exchange the states (phase levels and positions), where each robot receives messages from all others. The signaling effort depends on the natural frequency of oscillations. In our experiments, each robot sends eight messages per second (we use $\omega = \frac{8}{L}\,\mathrm{s}^{-1}$).

The total number of agents and the number of agents with the same phase level are unknown to the robots. For an update of phase correction and velocity, the number of messages received during the last oscillation cycle is assumed to be the number of agents. Therefore, if the messages are significantly delayed or dropped, it can have an impact on the convergence and stability of the desired pattern.

The same parameters as in the simulation are used: $T = 0.125\,\mathrm{s}$, $\epsilon_d = 0.1\,\mathrm{m}$, $d = 0.2\,\mathrm{m}$, and $v^{\mathrm{maxR}} = 0.2\,\mathrm{m/s}$.

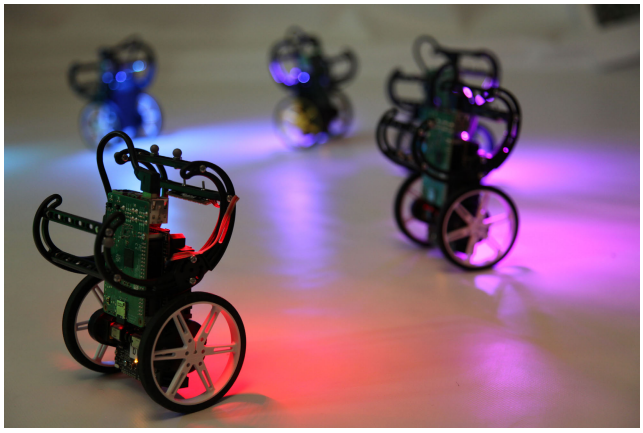


FIGURE 9. ***Static phase wave*: Balboa robots forming the pattern.**

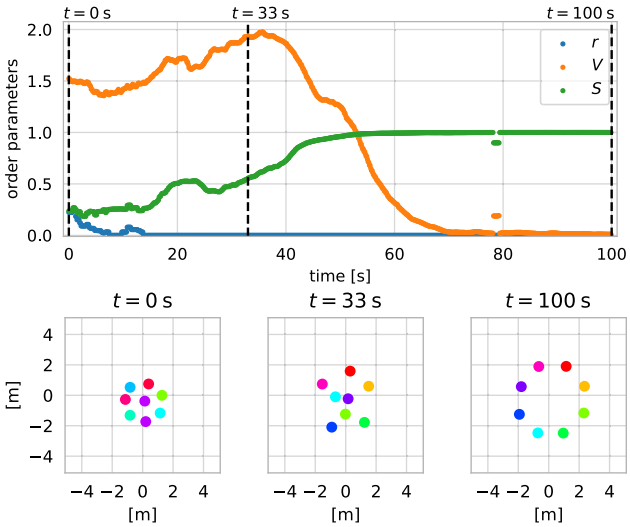


FIGURE 10. ***Static phase wave*: Convergence of order parameters and snapshots of the system showing the process of pattern forming with Balboa robots.**

The Balboas acting as sandsbots are able to successfully form the patterns. A snapshot from a system forming the static phase wave is shown in Figure 9. The convergence of order parameters for this pattern is presented in Figure 10. The pattern might get disturbed due to the non-deterministic communication delays and message drops (as happened at 78 s), but it quickly recovers.

VII. CONCLUSION AND OUTLOOK

Our time-discrete model for "sync and swarm" suited for multi-robot and drone systems successfully creates the emergent space-time patterns of swarmalator theory [3].

TABLE 1. Sandsbots — challenges.

Swarmalator model	Problem in robotic system	Solution in sandsbot model
Continuous coupling	Continuous coupling is not possible to achieve in a robotic system. Robots need to repeatedly exchange information about their positions and phases. Straightforward discretization with limited update rate causes physical oscillations and can lead to destabilization of static states.	Sandsbots are using time-discrete coupling with synchronized updates. This helps them to determine the exact states of the other agents at the time when the new update is calculated. In order to remove the physical oscillations and stabilize static states we use bounded attraction and repulsion functions and based on them dynamically limit the speed of sandsbots.
Instant coupling	The swarmalator model assumes that the state of other agents is known instantly, without any delays. If this information is delayed severe oscillations of positions can occur, destabilizing the system. Additionally, if swarmalators have natural frequency ($\omega \neq 0$) we have observed that using delayed states of other swarmalators causes static phase wave to rotate.	The moments when sandsbots are calculating updates, sending their states and applying their updates are synchronized. Additionally, each agent transmits its predicted state, from the moment when all agents apply their calculated updates. This guarantees that all agents are using the same data and the system behavior is deterministic. Agent's phase is discretized which helps in the situations when the inter-agent synchronization is imperfect.
Pattern depending on the initial conditions	1) In the swarmalator model in static phase wave there are no phase interactions between agents. This means that they are just sorting their initial phases. This state would look like static sync if swarmalators start with synchronized phases, not uniformly distributed. In robotic system with 10 to 20 agents drawing initial phases from uniform distribution leads to asymmetric state, because such a small sample is not uniform. 2) The exact number of clusters in splintered phase wave is not known. There is only a heuristic to estimate in which range the number of clusters will be. This number of clusters typically is too high compared to the number of agents we are considering, so this state cannot be observed in a small robotic swarm. For robotic applications it is of interest to know the number of clusters exactly and have the possibility to control it.	In sandsbot model we are using time coordination model to exactly control the number of clusters in a state or form splay state in the phase domain for static phase wave. Additionally, coupling of this model with position helps to minimize the traveled distance, because close neighbours tend to adjust their phases instead of going to the position fitting to the preassigned phase. Both states: static and splintered phase wave can be obtained independently of the phases in the initial state.
Point-shaped, freely moving particles	The swarmalator model treats agents as points; this approach would cause robots to collide. Additionally, robots are not able to perform unrestricted movement.	In sandsbot model we treat robots like disks of given size and constrain minimal distance between them. This helps us to avoid collisions. For each kind of robots and in the simulation we limit the maximum speed of movement. In [4] we also take into account kinematic constraints of movement for ground robots.
Global interactions	The swarmalator model assumes global, full knowledge of other agents' states. In the robotic system this might not be achievable. Some messages might be dropped or robots can be out of range.	In this work we assume that robots are working in proximity, nevertheless, we have done some initial analysis of limited range of interactions in sync and swarm. With some initial conditions it leads to incomplete convergence, e.g., multiple clusters in static sync, band-type formation in static phase wave. These variations of the original state are undesirable in robotic system. In the future work, resistance of the model to message drops is of interest.

It features controlled versions of static async and splintered phase wave with a specified number of clusters. These modifications enable us to create predictable patterns which can be employed in specific missions. Sending synchronized periodic updates makes the system robust when messages get delayed.

Future work on sandsbots includes the analysis of stability and robustness against message losses and the design of algorithms for automatic choice of parameters (e.g., to achieve a desired pattern size for certain applications). Another direction is to apply the concept of mode switching [29] for coupled temporal-spatial coordination.

APPENDIX
PROBLEMS WITH ROBOTIC APPLICATIONS OF SWARMALATOR MODEL

See Table 1.

REFERENCES

[1] R. E. Mirollo and S. H. Strogatz, "Synchronization of pulse-coupled biological oscillators," *SIAM J. Appl. Math.*, vol. 50, no. 6, pp. 1645–1662, Dec. 1990.

[2] C. W. Reynolds, "Flocks, herds and schools: A distributed behavioral model," in *Proc. 14th Annu. Conf. Comput. Graph. Interact. Techn. (SIGGRAPH)*, 1987, pp. 25–34.

[3] K. P. O'Keeffe, H. Hong, and S. H. Strogatz, "Oscillators that sync and swarm," *Nature Commun.*, vol. 8, no. 1, p. 1504, 2017.

[4] A. Barciś, M. Barciś, and C. Bettstetter, "Robots that sync and swarm: A proof of concept in ROS 2," in *Proc. Int. Symp. Multi-Robot Multi-Agent Syst. (MRS)*, Aug. 2019, pp. 98–104.
[5] A. Barciś and C. Bettstetter, "Beyond sync: Distributed temporal coordination and its implementation in a multi-robot system," in *Proc. IEEE 13th Int. Conf. Self-Adapt. Self-Organizing Syst. (SASO)*, Jun. 2019, pp. 88–96.
[6] F. Perez-Diaz, R. Zillmer, and R. Groß, "Firefly-inspired synchronization in swarms of mobile agents," in *Proc. Int. Conf. Auto. Agents Multiagent Syst.*, 2016, pp. 279–286.
[7] G. Brandner, U. Schilcher, and C. Bettstetter, "Firefly synchronization with phase rate equalization and its experimental analysis in wireless systems," *Comput. Netw.*, vol. 97, pp. 74–87, Mar. 2016.
[8] V. Trianni and S. Nolfi, "Self-organizing sync in a robotic swarm: A dynamical system view," *IEEE Trans. Evol. Comput.*, vol. 13, no. 4, pp. 722–741, Aug. 2009.
[9] M. Hartbauer and H. Römer, "A novel distributed swarm control strategy based on coupled signal oscillators," *Bioinspiration Biomimetics*, vol. 2, no. 3, pp. 42–55, 2007.
[10] A. L. Christensen, R. O'Grady, and M. Dorigo, "From fireflies to fault-tolerant swarms of robots," *IEEE Trans. Evol. Comput.*, vol. 13, no. 4, pp. 754–766, Aug. 2009.
[11] N. Bezzo, P. J. Cruz, F. Sorrentino, and R. Fierro, "Decentralized identification and control of networks of coupled mobile platforms through adaptive synchronization of chaos," *Phys. D: Nonlinear Phenomena*, vol. 267, pp. 94–103, Jan. 2014.
[12] T. Schmickl and H. Hamann, "BEECLUST: A swarm algorithm derived from honeybees: Derivation of the algorithm, analysis by mathematical models, and implementation on a robot swarm," in *Bio-Inspired Computing and Networking*, Y. Xiao, Ed. Boca Raton, FL, USA: CRC Press, 2016, pp. 95–137.
[13] G. Vásárhelyi, C. Virágh, G. Somorjai, T. Nepusz, A. E. Eiben, and T. Vicsek, "Optimized flocking of autonomous drones in confined environments," *Sci. Robot.*, vol. 3, no. 20, Jul. 2018, Art. no. eaat3536.
[14] Y. Quan Chen and Z. Wang, "Formation control: A review and a new consideration," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Aug. 2005, pp. 3181–3186.
[15] Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 1991, pp. 1398–1404.
[16] L. Barnes, M. Fields, and K. Valavanis, "Unmanned ground vehicle swarm formation control using potential fields," in *Proc. Medit. Conf. Control Autom.*, Jun. 2007, pp. 1–8.
[17] V. Gazi, "Swarm aggregations using artificial potentials and sliding-mode control," *IEEE Trans. Robot.*, vol. 21, no. 6, pp. 1208–1214, Dec. 2005.
[18] R. C. Fetecau, Y. Huang, and T. Kolokolnikov, "Swarm dynamics and equilibria for a nonlocal aggregation model," *Nonlinearity*, vol. 24, no. 10, pp. 2681–2716, Oct. 2011.
[19] V. Gazi and K. M. Passino, "Stability analysis of swarms," *IEEE Trans. Autom. Control*, vol. 48, no. 4, pp. 692–697, Apr. 2003.
[20] H. Tanner, A. Jadbabaie, and G. Pappas, "Stable flocking of mobile agents, Part I: Fixed topology," in *Proc. 42nd IEEE Int. Conf. Decis. Control*, Dec. 2003, pp. 2010–2015.
[21] L. Armijo, "Minimization of functions having lipschitz continuous first partial derivatives," *Pacific J. Math.*, vol. 16, no. 1, pp. 1–3, Jan. 1966.
[22] R. Sepulchre, D. Paley, and N. Leonard, "Collective motion and oscillator synchronization," in *Cooperative Control. Lecture Notes in Control and Information Science*, vol. 309, V. Kumar, N. Leonard, and A. S. Morse, Eds. Berlin, Germany: Springer, 2005, pp. 189–205.
[23] R. Sepulchre, D. A. Paley, and N. E. Leonard, "Stabilization of planar collective motion: All-to-All communication," *IEEE Trans. Autom. Control*, vol. 52, no. 5, pp. 811–824, May 2007.
[24] Y. Kuramoto, *Chemical Oscillations, Waves, and Turbulence*. New York, NY, USA: Dover, Aug. 2003.
[25] H. Gao and Y. Wang, "A pulse-based integrated communication and control design for decentralized collective motion coordination," *IEEE Trans. Autom. Control*, vol. 63, no. 6, pp. 1858–1864, Jun. 2018.
[26] K. P. O'Keeffe, J. H. M. Evers, and T. Kolokolnikov, "Ring states in swarmalator systems," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 98, no. 2, Aug. 2018, Art. no. 022203.
[27] K. P. O'Keeffe and C. Bettstetter, "A review of swarmalators and their potential in bio-inspired computing," *Proc. SPIE*, vol. 10982, May 2019, Art. no. 109822E.
[28] D. A. Paley, N. E. Leonard, and R. Sepulchre, "Oscillator models and collective motion: Splay state stabilization of self-propelled particles," in *Proc. 44th IEEE Conf. Decis. Control*, Dec. 2005, pp. 3935–3940.
[29] X. Yang, X. Li, J. Lu, and Z. Cheng, "Synchronization of time-delayed complex networks with switching topology via hybrid actuator fault and impulsive effects control," *IEEE Trans. Cybern.*, vol. 50, no. 9, pp. 4043–4052, Sep. 2020.

AGATA BARCIŚ received the M.Eng. degree in control engineering and robotics from the Wrocław University of Science and Technology, Poland. She is currently pursuing the Ph.D. degree with the Karl Popper School on Networked Autonomous Aerial Vehicles, University of Klagenfurt, Austria. Her main research interest is self-organization in robotics.

CHRISTIAN BETTSTETTER (Senior Member, IEEE) received the Dr.-Ing. degree *(summa cum laude)* in electrical and information engineering from Technical University of Munich, Germany. He is currently a Professor and the Head of the Institute of Networked and Embedded Systems, University of Klagenfurt, Austria. He is also the Scientific Director of Lakeside Labs, a research and innovation company, and a Faculty Member with the Karl Popper School on Networked Autonomous Aerial Vehicles. His research interests are in wireless connectivity and self-organization of networked systems with applications in telecommunications and robotics.