Decentralized Control of Minimalistic Robotic Swarms For Guaranteed Target Encapsulation

Himani Sinhmar and Hadas Kress-Gazit

Abstract— We propose a decentralized control algorithm for a minimalistic robotic swarm with limited capabilities such that the desired global behavior emerges. We consider the problem of searching for and encapsulating various targets present in the environment while avoiding collisions with both static and dynamic obstacles. The novelty of this work is the guaranteed generation of desired complex swarm behavior with constrained individual robots which have no memory, no localization, and no knowledge of the exact relative locations of their neighbors. Moreover, we analyze how the emergent behavior changes with different parameters of the task, noise in the sensor reading, and asynchronous execution.

I. INTRODUCTION

A swarm of robots is typically composed of simple individual robots with limited capabilities. Minimalistic swarm robotics [1] emphasizes the use of simple reactive robots which use pre-programmed behaviors, similar to reflexes, without maintaining any internal state. The simplicity of individual robots means they can be mass manufactured and can also be scaled to micro or nano-scale. This can be particularly relevant to nanomedical applications [2], [3] in which a single complex robot cannot be deployed due to space and energy constraints.

Developing decentralized control laws for robots in a swarm that *guarantee* the overall swarm behavior is a challenging task due to constraints such as limited computational power, imprecise locomotion, and the use of simple sensors. In this paper we present a discrete-time decentralized control algorithm for a robotic swarm with limited robot capabilities to search for and encapsulate targets in the environment, while avoiding collisions with static and dynamic obstacles. We are inspired by nanomedicine applications, such as a swarm of nano-robots searching for and encapsulating tumors by following a chemical gradient [4], [5]. In addition to the control, we provide bounds on parameters that will *guarantee* the swarm will achieve the task.

Related work: Work on minimalistic swarm robotics typically uses experiments or simulations to show the desired emergent behavior of the swarm given pre-programmed local rules. Using experiments with physical robots (e.g. [6], [7]), researchers have determined the optimal number of robots required for a swarm to complete a task; however, that work does not provide guarantees for achieving the desired behavior. In [8] and [9], simulations are used as a proof of concept to evaluate the aggregation and flocking capability

of a minimalistic algorithm. In [10], the authors propose a probabilistic model for studying the collaborative dynamics of robots pulling sticks, which depends on the geometry but is limited to events which are constant in space and time. In [11]–[16], the authors either use evolutionary algorithms or an exhaustive parameter grid search over the entire space of possible controllers and provide convergence guarantees, assuming no obstacles in the environment. Moreover, in these studies, the robots are either equipped with obstacle detecting range sensors or have an infinite sensing range. In [17] authors use genetic algorithm with novelty search to explore the space of controllers to determine the emergent behaviors possible given a limited set of robot capabilities.

Research in formal verification of swarms has used model checking techniques to prove properties of known swarm algorithms. In [18]–[20], the authors use temporal logic to formally specify the emergent behaviors of a robotic swarm and verify different properties of the swarm. While it provides formal guarantees, using model checking becomes intractable as the size of the system increases.

Recent studies have moved from minimalism and explored robots that use direct communication, broadcast information, and can learn and represent the environment. In [21], [22] an algorithm is proposed which is based on selective broadcasting of repulsion and attraction signals among swarm agents requiring limited direct communication between robots for a given swarm task. These studies also make use of extensive simulations and experiments for emergent behavioral analysis of the swarm without providing formal guarantees. The work in [23] introduced an automatic methodology to determine whether a swarm of robots with direct communication capabilities would display an emergent behavior irrespective of the number of agents present. In [24], [25] authors provide convergence guarantees on the emergent behavior of the swarm assuming that the robots have knowledge of the relative location of their neighbors.

Contribution: The novelty of this work is three fold: (i) we propose a correct-by-construction local control law for the robots that guarantees the emergence of the desired global behavior under bounds we compute on the maximum number of robots that are required for encapsulating a target, the maximum step size of a robot, and the minimum number of sensors on a robot, (ii) we consider minimalist robots that have no memory, no self-localization ability, do not know the relative location of their neighbors, no explicit communication ability, and are only equipped with omni-directional sensors and signal emitters, and (iii) we analyze the robustness of the control algorithm to noise in the sensors

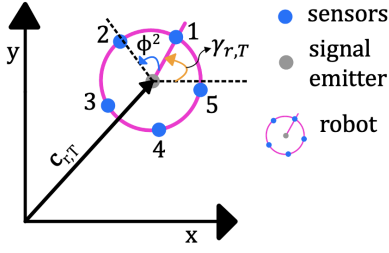Fig. 1: Robot model with $p = 5$.

and to asynchronous execution.

II. PRELIMINARIES

A. Environment and Robot model

Workspace: The robots operate in a continuous environment, $E \subseteq \mathbb{R}^2$ with a convex boundary. The environment has a fixed global frame, $\mathcal{F}$.

Target: A target $g = (\mathbf{c}_g, r_g)$ is a disk of radius r_g centered at $\mathbf{c}_g \in E$. $\mathcal{G}$ is a set of all targets contained in E.

Robot: A robot, $R = (\mathbf{c}_r, \gamma_r, r_r, p, Z)$, is modeled as a disk of radius r_r centered at $\mathbf{c}_r \in E$ with orientation $\gamma_r \in \mathbb{S}$, as shown in Fig. 1. Each robot is reactive and memoryless. It cannot localize itself and has no knowledge of the relative locations of other robots or targets. There is no explicit communication between the robots.

The robot is controlled through rotational and translational velocities [26] in a *turn-then-move* scheme . The kinematics of a robot are described by Eq. (1) which is a typical unicycle model [27]. At each time step, $\theta \in \mathbb{S}$ and $d \in \mathbb{R}^+$ are the control inputs corresponding to the angle turned and the distance moved by a robot. The maximum distance a robot can move at each time step is $d_{\max}$.

$$\gamma_{r,T} = \gamma_{r,T-1} + \theta$$
$$\mathbf{c}_{r,T} = \mathbf{c}_{r,T-1} + d[\cos\gamma_{r,T} \quad \sin\gamma_{r,T}]^T \quad (1)$$

Each robot has p omnidirectional sensors arranged on the boundary of the robot disk. Z denotes the set of measurements from all sensors. The angle between the kth sensor and the robot's x axis (heading direction) is denoted by ϕ^k $\forall k \in \{1 \cdots p\}$.

Signal sources: We consider three types of signals that a robot's sensor can detect: s_g from a point source at the center of a target, s_r from a point source at the center of the robot and s_e from a line source present on the entire environment boundary. The strength of any signal $s \in \{s_g, s_r, s_e\}$ as sensed by the robot's k^{th} sensor located at a distance d_j^k from signal source j is given by the function $B_s(d_j^k)$. Each signal source has a maximum influence distance β_s, such that $B_s(d_j^k) = 0$ $\forall d_j^k \geq \beta_s$. Every sensor can only sense the sum total of signal strength z_s^k which it receives from all signal sources of type s, as defined in Eq. (2). For the line source signal s_e, this summation becomes an integral over the boundary segment which lies inside the influence distance β_e.

$$z_s^k = \sum_j B_s(d_j^k) \quad (2)$$

To model realistic sensors, we add noise to z_s^k; specifically, we use the noise model in [28]. For example, a noise level of

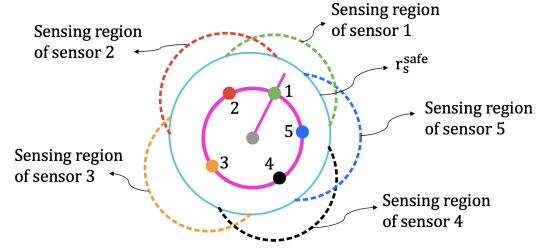Fig. 2: Possible asymmetric placement of sensors for $p = 5$. The sensing region of a sensor is equivalent to the influence region of a source. As can be seen, any source s, located at a distance of r_s^{safe} from the robot's center (cyan colored circle) will be inside the sensing region of at least one sensor.

15% of aggregated signal intensity is modeled as a normal distribution with a mean of 0 and a standard deviation of 0.15 as shown in Eq. (3)

$$z_s^k = (1 - n_s^k)\sum_j B_s(d_j^k), \; n_s^k \sim \mathcal{N}(0, 0.15^2) \quad (3)$$

To ensure that $z_s^k \geq 0$, we consider a truncated normal distribution, $-1 \leq n_s^k \leq 1$. For a robot, the kth sensor's reading consists of the tuple (z_g^k, z_r^k, z_e^k). Let $Z_g = \{z_g^1 \cdots z_g^p\}$, $Z_r = \{z_r^1 \cdots z_r^p\}$ and $Z_e = \{z_e^1 \cdots z_e^p\}$. Then the measurement set $Z = Z_g \cup Z_r \cup Z_e$. We define r_s^{safe} $\forall s \in \{g, r, e\}$ as the distance each robot must maintain from a source at all times.

We assume the sensors are arranged on the robot's boundary such that at least one sensor is in the influence region of a source, s, when the robot center is r_s^{safe} away from s. Note that we do not assume symmetric placement; Fig. 2 depicts one such valid, asymmetric placement. For the ease of exposition, in this paper we use a separation angle of $2\pi/p$ between sensors and indicate what change needs to be made in the case of asymmetric placement.

Target encapsulation: Let $\mathcal{A}_g = (\mathbf{c}_g, r_g^{\text{safe}}, r_g^{\text{encap}})$ be an annular region between two concentric circles of radius r_g^{safe} and r_g^{encap} centered at $\mathbf{c}_g$ such that $r_g^{\text{encap}} > r_g^{\text{safe}}$. The target $g \in \mathcal{G}$ is said to be **encapsulated** if the total number of robots currently present in the annular region, $\mathcal{A}_g$, is n_g. A robot is considered to be in the annular region of a target g if,

$$r_g^{\text{safe}} < \|\mathbf{c}_r - \mathbf{c}_g\| \leq r_g^{\text{encap}} \quad (4)$$

Assumption 1. Any two targets are at least $(2\beta_g + 2r_r + \epsilon)$ units apart, where ϵ is a small positive number. This ensures that a robot can sense at most one target at a time.

Assumption 2. When a target is encapsulated, it stops emitting a signal and emits a single burst of a shut off signal. The influence distance of this signal is limited to r_g^{encap} , and we assume that robots within the influence region, i.e. in the annular region $\mathcal{A}_g$, set $\boldsymbol{u} = 0$ thereafter but keep emitting their signal to ensure no collisions with other moving robots. This assumption emulates the behavior of nanorobots which would wrap around the tumor's surface to destroy it.

Assumption 3. The inverse of the signal function B_s exists and is known to the robots, i.e. given a signal strength reading of source s for the kth sensor, a robot can compute the distance $d_s^k = B_s^{-1}(z_s^k)$. Furthermore, we assume that the

signal strength strictly decreases with the radial distance from a source. Since each sensor only senses an aggregated signal, the computed distance d_s^k is typically not the true distance to a single source.

III. PROBLEM FORMULATION

Consider m targets in an environment E. Given the user-provided safe distance r_s^{safe} a robot R needs to maintain from a source, the number of robots n_g needed to encapsulate each target $g \in \mathcal{G}$, the total number of robots n such that $n \geq \sum_{g \in \mathcal{G}} n_g$, and the total number of sensors p on a robot, find a control law $\boldsymbol{u}(t)$ that will result in all targets $\mathcal{G}$ being encapsulated while ensuring the robots maintain r_s^{safe}.

In the following we derive such a control law while providing bounds on the parameters p, $d_{\max}$, β_r, r_g^{encap} and n_g which guarantee collision free encapsulation behavior.

IV. APPROACH

We design the control of a robot as a combination of three behaviors: (i) random walk when no target is sensed, (ii) moving towards a target when sensing one, and (iii) collision avoidance with the other robots, the targets and the environment boundary. The robot transitions between behaviors based on the signal thresholds $I_s^{\text{safe}} \, \forall s \in \{s_g, s_r, s_e\}$. In Section $IV-A$ we define virtual sources and use them later to compute I_s^{safe} based on r_s^{safe}. In Sections IV-B and IV-C we find control parameters for a single robot that guarantee it will move towards a target while avoiding collisions.

A. Virtual Sources

Since a robot's sensor measures only the total signal strength from nearby sources, it can neither figure out the relative location of the sources nor the total number of sources it is currently sensing. The same signal strength could correspond to a single source nearby or a cluster of sources farther away. Therefore, we define a *virtual* source at a radial distance from the sensor such that the signal strength from this source is equivalent to the total signal strength sensed by the sensor. Fig. 3 shows a single robot centered at point R and a *virtual* source centered at point S. The distance $d_s^k = B_s^{-1}(z_s^k)$ is known from the sensor reading z_s^k of the kth sensor (point A in Fig. 3), and S is located on a circle centered at point A with radius d_s^k. From the geometry of $\triangle SRA$ we have,

$$\overline{RS} = r_r\cos(\angle SRA) + \sqrt{(d_s^k)^2 - r_r^2\sin^2(\angle SRA)} \quad (5)$$

Assuming sensor k is receiving the strongest signal from source s, i.e. $z_s^k \geq z_s^l, \forall l \neq k$, we are interested in finding
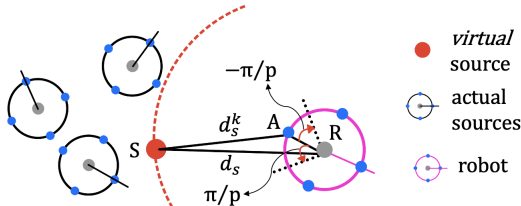

Fig. 3: *Virtual* source S located d_s^k from the kth sensor and d_s from robot center R such that $\angle SRA \in [-\pi/p, \ \pi/p]$.

the shortest possible distance d_s between the robot and the *virtual* source. We will use that to determine the maximum distance the robot can safely move. Now, if the *virtual* source is placed such that $\angle SRA > \pi/p$ in either direction of RA, then the maximum signal reading would be seen at $(k \pm 1)^{\text{th}}$ sensor and not at this sensor. This restricts the range of possible directions in which the *virtual* source can be located with respect to the robot's center to $\angle SRA \in [-\pi/p, \ \pi/p]$. It can then be seen that the source will be closest to the robot's center when $\angle SRA = \pm\pi/p$ and farthest when $\angle SRA = 0$. Substituting $\angle SRA = \pi/p$ and $\overline{RS} = d_s$ in the above equation we have,

$$d_s = r_r\cos(\pi/p) + \sqrt{(d_s^k)^2 - r_r^2\sin^2(\pi/p)} \quad (6)$$

Asymmetric sensor placement: Replace π/p in Eq. (6) with half of the maximum angle that the kth sensor makes with either of its adjacent sensors.

B. Target attraction

We use the Lyapunov stability theorem [29] to find the control parameters d and θ such that a robot moves towards a target $g \in \mathcal{G}$ when sensing it. Let $V = \|\mathbf{c}_g - \mathbf{c}_{r,T}\|^2 > 0$ be the candidate Lyapunov function, then $\mathbf{c}_g$ is stable if $V_{T+1} \leq V_T$, that is, $\|\mathbf{c}_g - \mathbf{c}_{r,T+1}\|^2 \leq \|\mathbf{c}_g - \mathbf{c}_{r,T}\|^2$. Using Eq. (1),

$$\|\mathbf{c}_g - \mathbf{c}_{r,T} - \mathbf{u}\|^2 \leq \|\mathbf{c}_g - \mathbf{c}_{r,T}\|^2$$

Let γ_g be the angle between the vectors $(\mathbf{c}_g - \mathbf{c}_{r,T})$ and $\mathbf{u}$, and $d = \|\mathbf{u}\|$ then,

$$\|\mathbf{c}_g - \mathbf{c}_{r,T}\|^2 + d^2 - 2d \|\mathbf{c}_g - \mathbf{c}_{r,T}\| \cos\gamma_g \leq \|\mathbf{c}_g - \mathbf{c}_{r,T}\|^2 \quad (7)$$

$$d^2 - 2d \|\mathbf{c}_g - \mathbf{c}_{r,T}\| \cos\gamma_g \leq 0 \quad (8)$$

Ignoring the unlikely, perfectly symmetric scenario where 2 sensors receive the maximum intensity from a target, let k be the index of the sensor such that $z_g^k > z_g^l, \forall l \neq k$. Then the direction of the target with respect to the robot's center is within the angular range $[\phi^k - \pi/p, \ \phi^k + \pi/p]$ as explained in Section IV-A. This implies that $\gamma_g \in [\theta - (\phi^k - \pi/p), \ \theta - (\phi^k + \pi/p)]$ where θ is the direction of the robot motion. The necessary condition to satisfy Eq. (8) is that $\cos\gamma_g \geq 0$, that is, $\gamma_g \in [3\pi/2, \ 2\pi] \cup [0, \ \pi/2]$. Using these two conditions, the angular range of possible control directions θ is given by Eq. (9). Fig. 4 shows an example of the angular range Θ_g^{att} for a robot with $p = 5$ as computed by its sensor closest to the target.

$$\Theta_g^{\text{att}} = [\phi^k + \pi/p + 3\pi/2, \ \phi^k - \pi/p + \pi/2] \quad (9)$$

Let d_g be the estimate of $\|\mathbf{c}_g - \mathbf{c}_{r,T}\|$ obtained using z_g^k, then from Eq. (8), $d \leq 2d_g\cos\gamma_g$. In Algorithm 1, **DistAttractTarget** computes the maximum possible value of d such that, $0 \leq d \leq \min\{2d_g\cos\gamma_g, \ d_{\max}\}$. **Asymmetric sensor placement:** Replace $-\pi/p$ and π/p in Eq. (9) with half of the angle that the kth sensor makes with sensor $k - 1$ and sensor $k + 1$ respectively (assuming counterclockwise ordering of sensors as shown in Fig. 1).
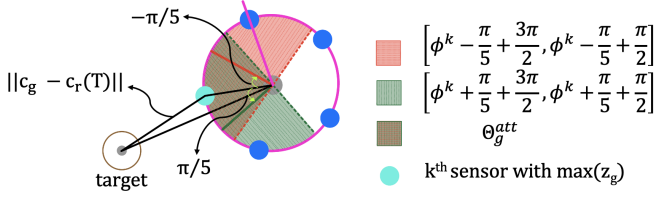
Fig. 4: The direction of motion θ for a robot should be within the angular range Θ_g^{att} for it to move towards a target.

C. Collision Avoidance

To avoid collisions with a source $s \in \{g, r, e\}$, we set signal thresholds I_s^{safe} such that the robot triggers collision avoidance behavior before the distance between a source and a robot's center, as estimated from a sensor's reading, is equal to r_s^{safe}. Since r_s^{safe} is defined between the robot's center and the source we set $d_s = r_s^{\text{safe}}$ in Eq. (6), to obtain $d_s^k = \sqrt{(r_s^{\text{safe}})^2 + r_r^2 - 2r_r r_s^{\text{safe}}\cos(\pi/p)}$. Furthermore, to account for a scenario where the distance between the source and the robot is just marginally greater than r_s^{safe}, we add the maximum distance $d_{\max}$ the robot can move at this time step to d_s^k. Then the threshold strength $I_s^{\text{safe}}, s \in \{g, r, e\}$ is given by Eq. (10).

$$I_s^{\text{safe}} = B_s\left(d_{\max} + \sqrt{(r_s^{\text{safe}})^2 + r_r^2 - 2r_r r_s^{\text{safe}}\cos(\pi/p)}\right) \tag{10}$$

Asymmetric sensor placement: Replace π/p in Eq. (10) with half of the maximum angle between two adjacent sensors on the robot.

Collision avoidance with a static obstacle: Let $\mathbf{c}_o^{\text{static}}$ be the location of a static obstacle (target or boundary). To avoid collision, the robot's motion at time step T should be such that it does not move towards the obstacle, that is $\|\mathbf{c}_o^{\text{static}} - \mathbf{c}_{r,T+1}\| \geq \|\mathbf{c}_o^{\text{static}} - \mathbf{c}_{r,T}\|$.

Let k be the index of the sensor receiving the maximum intensity from the static source $s \in \{g, e\}$ such that $z_s^k \geq I_s^{safe}$ and $z_s^k > z_s^l, \forall l \neq k$. Then the direction of the obstacle with respect to the robot's center is within the angular range $[\phi^k - \pi/p, \ \phi^k + \pi/p]$ as explained in Section IV-A. Let γ_o be the angle between the vectors $(\mathbf{c}_o^{\text{static}} - \mathbf{c}_{r,T})$ and $\mathbf{u}$, that is $\gamma_o \in [\theta - (\phi^k - \pi/p), \ \theta - (\phi^k + \pi/p)]$. Now, using Eq. (1) to simplify the collision avoidance constraint we get,

$$d^2 - 2d\|\mathbf{c}_o^{\text{static}} - \mathbf{c}_{r,T}\|\cos\gamma_o \geq 0 \tag{11}$$

If γ_o is chosen such that $\cos\gamma_o \leq 0$, then Eq. (11) is always satisfied. Using both constraints, the angular range for θ to avoid static obstacles is given by Eq. (12).

$$\Theta_s^{\text{avo}} = [\phi^k + \pi/p + \pi/2, \ \phi^k - \pi/p + 3\pi/2] \tag{12}$$

Asymmetric sensor placement: Replace $-\pi/p$ and π/p in Eq. (12) with half of the angle that k^{th} sensor makes with sensor $k - 1$ and sensor $k + 1$ respectively (assuming counterclockwise ordering of sensors as shown in Fig. 1).

Collision avoidance with other robots: In the case of dynamic obstacles (i.e. other robots), the condition in Eq. (11) would not be sufficient for collision avoidance, since the other robot can move toward the robot. Furthermore, a robot might be surrounded by multiple moving robots, so the

control parameters should be chosen such that it avoids all of the nearby robots.

For any sensor k, its virtual source is located at a radial distance of $d_r^k = B_r^{-1}(z_r^k)$ within the angular range $[\phi^k - 2\pi/p, \ \phi^k + 2\pi/p]$ as shown in Fig. 5. At time step T, given
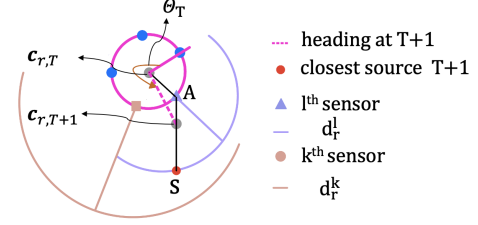

Fig. 5: The distance to move in θ direction is computed using the reading from k^{th} and l^{th} sensor.

the direction of motion θ (shown by the dashed magenta line), let k and l be the indices of the sensors that are closest to θ i.e. $\phi^k < \theta < \phi^l$ such that $d_r^l < d_r^k$. Then the distance that the robot can move is chosen such that it maintains a safe distance of r_r^{safe} from neighboring moving robots after moving d units in the direction of motion. At $T + 1$, the closest virtual source to the robot is at $\mathbf{S}$. The maximum distance that this virtual source could have moved at T is $d_{\max}$. To ensure safety, $\|\mathbf{c}_{r,T+1} - \mathbf{S}\| \geq r_r^{\text{safe}} + d_{\max}$. Using the geometry of $\triangle \mathbf{c}_{r,T} A \mathbf{c}_{r,T+1}$, we have

$$d_r^l - \sqrt{d^2 + r_r^2 - 2dr_r\cos(\phi^l - \theta)} \geq r_r^{\text{safe}} + d_{\max} \tag{13}$$

$$0 \leq d \leq r_r\cos(\phi^l - \theta) + \sqrt{(d_r^l - r_r^{\text{safe}} - d_{\max})^2 - r_r^2\sin^2(\phi^l - \theta)} \tag{14}$$

In Algorithm 1, the function **DistAvoDynObs** computes the maximum possible value of d from Eq. (14).

D. Control for Each Robot

Algorithm (1) describes the control generation for a robot in the swarm. If the total signal strength received by a robot from static sources $s \in \{g, e\}$ is greater than the preset safe threshold, i.e. $\max(Z_s) \geq I_s^{\text{safe}}$, the robot finds a direction of motion ($\theta \in \Theta_s^{\text{avo}}$) that maximizes the possible distance, d, such that it moves away from the source s while avoiding nearby moving robots (lines 2-3).

When the robot senses a target $g \in \mathcal{G}$ such that the maximum signal strength received is less than the safety threshold I_g^{safe}, it moves towards the target while avoiding collisions with nearby robots. The parameters d and θ for this behavior are chosen based on Section IV-B (lines 4-6).

If the computed distance d is zero, the robot chooses the control parameters based on the reading of sensor k receiving the minimum signal strength z_r^k (lines 7-10), as that is the safest direction to move in. When the robot is outside the influence of all targets, it performs a random walk while avoiding collisions with nearby robots (lines 11-17).

V. SAFETY AND LIVENESS GUARANTEES

In this section we assume noiseless sensors and derive constraints on different parameters, that if satisfied, guarantee

Algorithm 1: Control algorithm for a robot

Input : Z, B_s, p, r_s^{safe}, $\forall s \in \{r, g, e\}$
Output: d, θ
`// compute` Θ_g^{att}, Θ_e^{avo}, Θ_g^{avo}
1 **if** $max(Z_s) \geq I_s^{\text{safe}}$, $s \in \{g, e\}$ **then**
2     $\theta = \underset{\varphi \in \Theta_s^{\text{avo}}}{\arg\max} \texttt{DistAvoDynObs}(Z_r, B_r, \varphi, r_r^{\text{safe}})$
3     $d = \texttt{DistAvoDynObs}(Z_r, B_r, \theta, r_r^{\text{safe}})$
4 **else if** $0 < max(Z_g) < I_g^{\text{safe}}$ **then**
5     $\theta = \underset{\varphi \in \Theta_g^{\text{att}}}{\arg\max} \texttt{DistAvoDynObs}(Z_r, B_r, \varphi, r_r^{\text{safe}})$
6     $d = \min\big(\texttt{DistAvoDynObs}(Z_r, B_r, \theta, r_r^{\text{safe}}),$
       $\texttt{DistAttractTarget}(Z_g, B_g, \theta, r_g^{\text{safe}})\big)$
7   **if** $d = 0$ **then**
8         $k = \texttt{argmin}(Z_r)$
9         $\theta = \phi^k$
10        $d = \texttt{DistAvoidRobots}(Z_r, B_r, \theta, r_r^{\text{safe}})$
11 **else**
12     $\theta = \texttt{randsample}([0,\ 2\pi])$
13     $d = \texttt{DistAvoDynObs}(Z_r, B_r, \theta, r_r^{\text{safe}})$
14   **if** $d = 0$ **then**
15         $k = \texttt{argmin}(Z_r)$
16         $\theta = \phi^k$
17        $d = \texttt{DistAvoidRobots}(Z_r, B_r, \theta, r_r^{\text{safe}})$

---

that there are no collisions, the robots are never stuck in a deadlock and all targets in the environment are encapsulated. In Section VI we analyze how sensor noise and asynchronous control execution affect these guarantees and lead to interesting emergent behaviors.

A. Safety: Collision avoidance

Lemma V.1. *The estimate of the relative distance d_s between a virtual source and robot's center in Eq. (6) is always less than or equal to the distance from the actual closest source.*

Proof. A sensor receives the sum total of signal strengths from all nearby sources, which is always greater than or equal to the signal strength from a single source because $B_s(d)$ is always positive. From Assumption (3) and Eq. (2), the radial distance d_s^k of the *virtual* source from the kth sensor decreases as the sensor reading z_s^k increases. Hence, the radial distance d_s^k is always equal to (if there is a single source) or less than (if there are multiple nearby sources) the radial distance from the actual closest source. As detailed in Section IV-A and Eq. (6), a robot always chooses the minimal possible radial distance d_s from the virtual source. $\square$

Lemma V.2. *If every robot in the swarm implements a local behavior, as outlined in the Algorithm 1, such that the following requirements hold, then a robot always maintains a given safe distance r_s^{safe} from a source $s \in \{g, r, e\}$:*
1) *The influence distance of a source $\beta_s \geq B_s^{-1}(I_s^{\text{safe}})$*
2) *In the initial state, a robot is at least r_s^{safe} units away from every source*

Proof. Condition (1) ensures that the maximum influence distance of a source is at least r_s^{safe}. Using Lemma V.1 and Eq. (10), collision avoidance behavior (IV-C) is always

triggered for a robot before the distance between a source and a robot's center, as estimated from a sensor's reading, is equal to r_s^{safe}. As elaborated in Section IV-C, a robot always either moves in a direction with no obstacles or moves a distance d such that a minimum distance of r_s^{safe} is maintained from the *virtual* source. Condition (2) enforces that the swarm is safe at the initial time step. $\square$

B. Absence of deadlocks

A deadlock occurs in a swarm if two or more robots are in a configuration where none of them can move, i.e. $d = 0$.

Lemma V.3. *No deadlocks can occur in a swarm if,*
1) *The total number of robots in the swarm is such that they can be placed in a configuration where any two robots are at least $(\beta_r + r_r)$ units apart.*
2) *The maximum influence distance of a robot's source β_r satisfies $B_r^{-1}(I_r^{\text{safe}}) + d_{\max} < \beta_r < r_r^{\text{safe}} + r_r\cos(\pi/p)$*
3) *The maximum distance a robot can move in a time step $d_{\max} < \frac{r_r^{\text{safe}} + r_r\cos(\pi/p) - \sqrt{(r_r^{\text{safe}})^2 + r_r^2 - 2r_r r_r^{\text{safe}}\cos(\pi/p)}}{2}$*
4) *The total number of sensors on a robot is $p \geq 3$*

Proof. To ensure safety, the influence region of a robot's source should be large enough so that Lemma V.2 is satisfied. In a scenario where a robot is marginally outside β_r of a nearby robot and moves $d_{\max}$, we get $\beta_r > B_r^{-1}(I_r^{\text{safe}}) + d_{\max}$. Furthermore, the influence distance should be small enough such that when two robots are at a relative distance of r_r^{safe}, at least one sensor on each robot is outside the influence region of the other robot. This, along with condition (1), ensures that there always exists at least one robot in the swarm which has $\mathbf{u} \neq 0$ in an obstacle free direction.

Fig. 6 shows two robots that are r_r^{safe} apart. From geometry, we get $\beta_r < r_r^{\text{safe}} + r_r\cos(\pi/p)$. These constraints on β_r give us an upper bound on the maximum step size of a robot,

$$B_r^{-1}(I_r^{\text{safe}}) + d_{\max} < r_r^{\text{safe}} + r_r\cos(\pi/p) \quad (15)$$

Using Eq. (10) we have,

$$2d_{\max} + \sqrt{(r_r^{\text{safe}})^2 + r_r^2 - 2r_r r_r^{\text{safe}}\cos(\pi/p)} \\ < r_r^{\text{safe}} + r_r\cos(\pi/p) \quad (16)$$

$$d_{\max} < \frac{r_r^{\text{safe}} + r_r\cos(\pi/p)}{2} \\ - \frac{\sqrt{(r_r^{\text{safe}})^2 + r_r^2 - 2r_r r_r^{\text{safe}}\cos(\pi/p)}}{2} \quad (17)$$

Fig. 7 shows how $d_{\max}$ (as a function of r_r) changes with the number of sensors p. As can be seen from Eq. (17), as p grows, $d_{\max}$ approaches r_r. Furthermore, we require $p \geq 3$ otherwise $d_{\max} < 0$ for any given $r_r^{\text{safe}} > 0$. $\square$

Asymmetric sensor placement: Replace π/p in Eq. (15) with half of the maximum angle between two adjacent sensors on a robot.
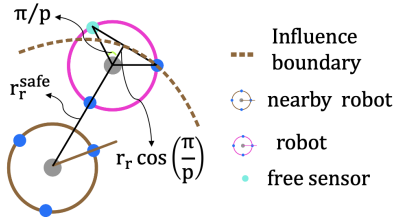
Fig. 6: When two robots are r_r^{safe} distance apart at least one sensor is outside the influence region β_r.
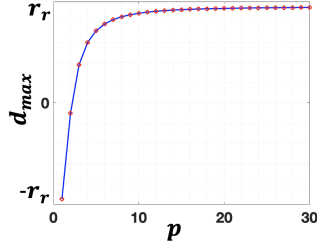

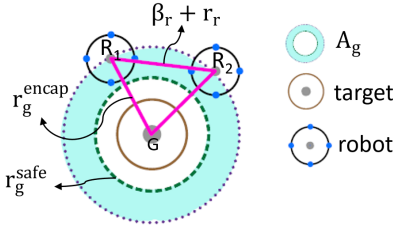Fig. 7: $d_{\max}$ as a function of p for equally spaced sensors.

Fig. 8: Geometry of a target and two robots to compute an upper bound on n_g.

C. Liveness: Encapsulating all targets

Lemma V.4. *A robot performing a random walk in a bounded environment will always eventually explore the entire area [30].*

Lemma V.5. *For any random initial condition such that,*

1) *the number of robots n_g required to encapsulate a target g satisfies,*

$$n_g \leq \frac{2\pi}{\cos^{-1}\left(1 - \frac{(\beta_r + r_r)^2}{2(r_g^{\text{encap}})^2}\right)} = n_0 \qquad (18)$$

2) *the outer radius of the annular region $\mathcal{A}_g$, $r_g^{\text{encap}} \geq r_g^{\text{safe}} + 2d_{\max}$.*

all $g \in \mathcal{G}$ will eventually be encapsulated.

Proof. The maximum number of robots, n_0, that can be present simultaneously in the annular region $\mathcal{A}_g$ to satisfy Eq. (4) should be such that they are outside each other's influence region. That is, the minimum relative distance between any two consecutive robots is $\beta_r + r_r$. This, along with condition (2), ensures that there exists a configuration where the robots encapsulating a target are in the annular region without any chattering. From Fig. 8, $\angle R_2 G R_1 = cos^{-1}\left(1 - \frac{(\beta_r + r_r)^2}{2(r_g^{\text{encap}})^2}\right)$ is given by the cosine rule of triangles. We define $n_0 = \frac{2\pi}{\angle R_2 G R_1}$. Since in the annular region a

robot oscillates between getting attracted to a target and maintaining a distance of r_g^{safe} from it, the upper bound on r_g^{encap} is such that Eq. (4) can be satisfied despite these oscillations.

Let the number of robots specified to encapsulate a target n_g be less than n_0. At any time step T, if there are less than n_g robots in $\mathcal{A}_g$, a robot in the influence of a target always finds either an obstacle free direction to move towards the target or it finds a non-zero distance to move in the direction of the sensor receiving the minimum signal from nearby robots (Lemma V.3). This behavior leads to an increment in the Lyapunov function of the robot but ensures that the robot does not get stuck in a local minima caused by an obstacle between itself and the target. When $n_g > n_0$, there is a dynamic equilibrium of robots near the target such that at least n_0 robots are almost always present in $\mathcal{A}_g$.

From assumption (2), when a target is encapsulated, it stops emitting a signal and all the robots in $\mathcal{A}_g$ stop moving. The robots that were outside $\mathcal{A}_g$ but inside the target's influence region will transition into a random walk behavior. It follows from Lemma V.4 and Lemma V.3 that a swarm will always eventually encapsulate all the targets. $\square$

VI. ANALYSIS OF SWARM BEHAVIOR

In this section, we investigate how the parameters p, n_g, and noisy sensors affect the emergent behavior of the swarm. We used three metrics to compare the emergent behavior: the total time for encapsulating all targets $g \in \mathcal{G}$, the cumulative path length traveled by all robots, and the probability of target encapsulation while ensuring no collisions.

In the following, the environment consists of one target and the number of robots is $n = 10$. We fixed the total simulation time to be 3000 time steps and ran 100 simulations for each data point with the same initial conditions. This was done to study the effect of the parameters on the emergent behavior by keeping other conditions constant. The randomness in each simulation is due to the random choice of θ. Moreover, at each time step, all robots in the swarm move with different rotational and translational speeds where the translational speed is capped at $(d_{\max}/\Delta t)$.

Effect of the noisy sensors: If a measurement is noisy, it implies that the distance a robot estimates using **DistAvoDynObs** and **DistAttractTarget** is not accurate, which might cause collisions. To avoid collisions, a simple solution would be to tweak the safety thresholds defined in Section IV-C; however, this would require a bounded noise model with known bounds. We ran two experiments to analyze the affect of noise on the behavior.

Experiment 1: We did not modify the safety thresholds and all robots implemented algorithm 1 as is, with $p = 7$. In Fig. 9, we show how increasing the noise levels (Eq. 3) affects the total number of collisions observed between robots and static and dynamic obstacles. We repeated this experiment (100 simulations per noise level) for each of the following: (i) different white noise added to each sensor on every robot (ii) the same white noise added to all sensors on a robot (iii) the same white noise added to all sensors on all the
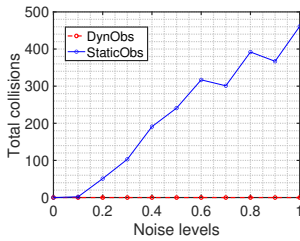
Fig. 9: Total collisions with dynamic and static obstacles; safety bounds assume noiseless measurements.

robots in the swarm. In all cases of added noise, we observed that the number of collisions with static obstacles (targets and environment boundary) increased with an increase in the noise levels. In contrast, at all noise levels there were no collisions among robots.

A possible hypothesis for this emergence is that our control algorithm ensures that robots in close vicinity move so as to avoid each other. For clusters of moving robots, the virtual source is already closer than the actual source, resulting in less sensitivity to noisy measurements. Such robustness to noise could be attributed to the fact that the randomness of white noise is averaged out to zero in the swarm. In case of static obstacles, since the source does not move away from the robot, the noise filtering is no longer two-sided and hence there is a higher probability of collisions.

Experiment 2: Since collisions with only static obstacles were observed, we truncated the noise on static signals to $-0.6 < n_s^k \leq 0.6 \ \forall s \in \{g, e\}, \ \forall k \in \{1 \cdots p\}$. We also changed the safety thresholds for static obstacles to $I_s^{\text{safe}} = I_s^{\text{safe}}(1 - 0.6) \forall s \in \{g, e\}$. This ensured that the collision avoidance behavior is triggered in the worst case scenario of a robot being located almost at r_s^{safe} of a static source but estimating it to be further away. A decrease in I_s^{safe} implies that r_s^{safe} increased, resulting in the increase of r_g^{encap} or the area of the annular ring in Fig. 8. This led to an increase in the bound on n_0 from Eq. (18). Given this more conservative bound, we did not observe any collisions with static or dynamic obstacles.

For all the remaining analysis below, we consider a noise level of 15% in the measurements and a truncated noise addition (within 2 standard deviations) for signals from static sources (targets and the boundary).

Effect of asynchronous control: In Fig. 10a we show how the frequency of the sensing and control updates affects the total time taken for encapsulation as compared to synchronous control. We set $n_g = 5$ and an initial update time of $t_0 = 0$ for all robots. For asynchronous control, the sensors update frequency for each robot was randomly chosen from $[1, 2, 3, 4]$ global time steps; it was 1 time step for synchronous control. Less frequent sensor and control updates of some robots in the swarm resulted in a higher total time taken for task completion. We observed a similar phenomenon in Fig. 10b where the sensor and control update frequency is the same (every 2 global time steps) for all robots, but they differ in their initial start time which
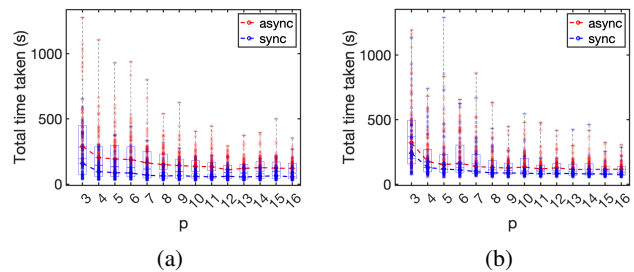

(a) (b)

Fig. 10: The total time taken for task completion as a function of p for asynchronous and synchronous control such that in (a) each robot has a different sensor update frequency with the same initial start time and in (b) each robot has a different initial start time with same sensor update frequency. The box plot shows median, 25th and 75th percentiles and the min/max values. The line connects the medians.
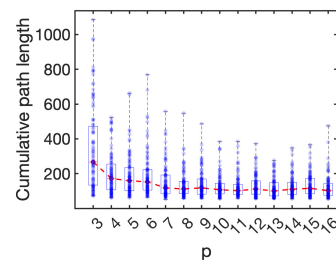

Fig. 11: Cumulative path length of all robots as a function of p. Distance is measured in robot diameter. The line connects the medians.

was randomly chosen from $\{0, 1, 2, 3, 4, 5\}$. For synchronous control, all robots started at $t_0 = 0$ with sensing and control updates every 2 global time steps.

Effect of the number of sensors p: In Fig. 11, we show how increasing the number of sensors on a robot affects the cumulative path length traveled by the robots for encapsulating the target. Here $n_g = 5$ and the control is synchronous. The cumulative path length decreased quickly as the number of sensors are increased. This is because with an increase in p, robots have an increased sense of directionality, resulting in less chattering due to simultaneous attraction and repulsion towards the target and nearby robots.

Effect of varying n_g: In Fig. 12a, we show how varying the number of robots required for target encapsulation n_g affects the total time taken for target encapsulation. The number of sensors is $p = 8$, and $n_0 = 6.9051$ implying $n_g \leq 6$ for guaranteed task completion . Fig. 12b shows the probability of task completion. As expected, the task is completed 100% of the time if $n_g \leq 6$, it is completed 26% of the time for $n_g = 7$, and is never completed within the time bound for $n_g > 7$. Hence, the guaranteed desired emergence only happens when the conditions in Section V are met.

Scalability and Emergent behaviors(video): In the supplemental video we demonstrate different emergent swarm behaviors created by different parameter choices. Our control algorithm is highly scalable, as shown in a large scale simulation of 25 targets and 200 robots. We also show task completion for asymmetric placement of sensors.
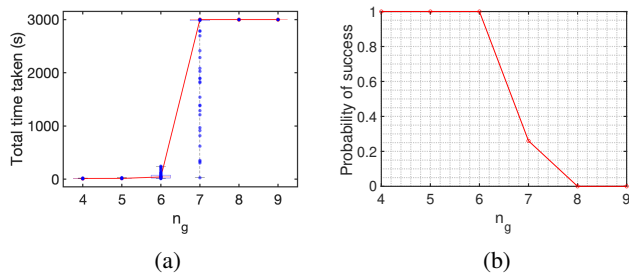
Fig. 12: (a) Total time taken for target encapsulation (with simulation time capped at 3000 time-steps) and (b) probability of success for target encapsulation as a function of n_g when $n_0 = 6.9051$.

VII. Conclusion

In this paper, we show how robots equipped with simple omnidirectional sensors and an isotropic signal emitter are capable of finding and encapsulating targets in the environment while avoiding both static and dynamic obstacles. Our decentralized controller is agnostic to the number of robots and targets in the environment. We presented a detailed analysis of the implemented decentralized controller and provided bounds on the maximum step size of a robot, minimum number of required sensors, and maximum number of robots required for encapsulation such that the desired emergent behavior is guaranteed. We further studied the effects of noise, uncertainties, and asynchronous control on the emergent behavior and observed an interesting phenomenon of robot-robot collision avoidance regardless of the (bounded) noise level. In future work, we will implement our algorithm on physical robots. We will analyze how the presence of other obstacles and targets moving in patterns affect the emergent behavior. We will also explore different target search strategies such as Lévy walk and correlated random walks for a swarm with no memory.

References

[1] A. J. C. Sharkey, "Swarm robotics and minimalism," *Connect. Sci*, vol. 19, p. 245–260, Sept. 2007.

[2] B. J. Nelson, I. K. Kaliakatsos, and J. J. Abbott, "Microrobots for minimally invasive medicine," *Annual Review of Biomedical Engineering*, vol. 12, no. 1, pp. 55–85, 2010. PMID: 20415589.

[3] R. Freitas, "Pharmacytes: an ideal vehicle for targeted drug delivery.," *Journal of nanoscience and nanotechnology*, vol. 6 9-10, pp. 2769–75, 2006.

[4] S. S. Andhari, R. D. Wavhale, K. D. Dhobale, B. V. Tawade, G. P. Chate, Y. N. Patil, J. J. Khandare, and S. S. Banerjee, "Self-propelling targeted magneto-nanobots for deep tumor penetration and ph-responsive intracellular drug delivery," *Scientific Reports*, vol. 10, p. 4703, Mar 2020.

[5] A. Cavalcanti, B. Shirinzadeh, R. A. F. Jr, and T. Hogg, "Nanorobot architecture for medical target identification," *Nanotechnology*, vol. 19, p. 015103, nov 2007.

[6] R. Beckers, O. E. Holland, and J.-L. Deneubourg, *Fom Local Actions to Global Tasks: Stigmergy and Collective Robotics*, pp. 1008–1022. Dordrecht: Springer Netherlands, 2000.

[7] C. Kube and E. Bonabeau, "Cooperative transport by ants and robots," *Robotics and Autonomous Systems*, vol. 30, no. 1, pp. 85–101, 2000.

[8] C. Moeslinger, T. Schmickl, and K. Crailsheim, "A minimalist flocking algorithm for swarm robots," in *Advances in Artificial Life. Darwin Meets von Neumann* (G. Kampis, I. Karsai, and E. Szathmáry, eds.), (Berlin, Heidelberg), pp. 375–382, Springer Berlin Heidelberg, 2011.

[9] O. Soysal and E. Sahin, "Probabilistic aggregation strategies in swarm robotic systems," in *Proceedings 2005 IEEE Swarm Intelligence Symposium, 2005. SIS 2005.*, pp. 325–332, 2005.

[10] A. Ijspeert, A. Martinoli, A. Billard, and L. Gambardella, "Collaboration through the exploitation of local interactions in autonomous collective robotics: The stick pulling experiment," *Autonomous Robots*, vol. 11, pp. 149–171, 09 2001.

[11] D. St-Onge, C. Pinciroli, and G. Beltrame, "Circle formation with computation-free robots shows emergent behavioural structure," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5344–5349, 2018.

[12] P. Mitrano, J. Burklund, M. Giancola, and C. Pinciroli, "A minimalistic approach to segregation in robot swarms," in *2019 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, pp. 105–111, Aug 2019.

[13] A. Ozdemir, "Synthesis and analysis of minimalist control strategies for swarm robotic systems." April 2020.

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

[15] M. Gauci, J. Chen, T. J. Dodd, and R. Groß, "Evolving aggregation behaviors in multi-robot systems with binary sensors," in *Distributed Autonomous Robotic Systems* (M. Ani Hsieh and G. Chirikjian, eds.), (Berlin, Heidelberg), pp. 355–367, Springer Berlin Heidelberg, 2014.

[16] M. Gauci, J. Chen, W. Li, T. J. Dodd, and R. Groß, "Clustering objects with robots that do not compute," in *AAMAS*, 2014.

[17] D. S. Brown, R. Turner, O. Hennigh, and S. Loscalzo, "Discovery and exploration of novel swarm behaviors given limited robot capabilities," in *DARS*, 2016.

[18] C. Dixon, A. Winfield, and M. Fisher, "Towards temporal verification of emergent behaviours in swarm robotic systems," in *Towards Autonomous Robotic Systems* (R. Groß, L. Alboul, C. Melhuish, M. Witkowski, T. J. Prescott, and J. Penders, eds.), (Berlin, Heidelberg), pp. 336–347, Springer Berlin Heidelberg, 2011.

[19] A. Winfield, J. Sa, C. Fernandez Gago, C. Dixon, and M. Fisher, "On formal specification of emergent behaviours in swarm robotic systems," *International Journal of Advanced Robotic Systems*, vol. 2, 12 2005.

[20] M. Brambilla, A. Brutschy, M. Dorigo, and M. Birattari, "Property-driven design for robot swarms: A design method based on prescriptive modeling and model checking," *ACM Trans. Auton. Adapt. Syst.*, vol. 9, Dec. 2014.

[21] A. R. Shirazi and Y. Jin, "Regulated morphogen gradients for target surrounding and adaptive shape formation," *IEEE Transactions on Cognitive and Developmental Systems*, pp. 1–1, 2020.

[22] S. O. Obute, M. R. Dogar, and J. H. Boyle, "Simple swarm foraging algorithm based on gradient computation," *CoRR*, vol. abs/1906.07030, 2019.

[23] P. Kouvaros and A. Lomuscio, "Verifying emergent properties of swarms," in *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, p. 1083–1089, AAAI Press, 2015.

[24] M. Coppola, J. Guo, E. Gill, and G. C. H. E. de Croon, "Provable self-organizing pattern formation by a swarm of robots with limited knowledge," *Swarm Intelligence*, vol. 13, pp. 59–94, Mar 2019.

[25] V. Gazi and K. Passino, "Stability analysis of social foraging swarms," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 34, no. 1, pp. 539–557, 2004.

[26] S.-i. Azuma, K. Owaki, N. Shinohara, and T. Sugie, "Performance analysis of chemotaxis controllers: Which has better chemotaxis controller, escherichia coli or paramecium caudatum?," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 13, no. 4, pp. 730–741, 2016.

[27] R. M. Murray, S. S. Sastry, and L. Ze-xiang, "A mathematical introduction to robotic manipulation," 1994.

[28] S. O. Obute, P. Kilby, M. R. Dogar, and J. H. Boyle, "Repatt: Achieving swarm coordination through chemotaxis," in *2020 IEEE 16th International Conference on Automation Science and Engineering (CASE)*, pp. 1307–1312, 2020.

[29] H. K. Khalil, *Nonlinear systems; 3rd ed.* Upper Saddle River, NJ: Prentice-Hall, 2002. The book can be consulted by contacting: PH-AID: Wallet, Lionel.

[30] S. Popov, *Recurrence of two-dimensional simple random walk*, p. 8–32. Institute of Mathematical Statistics Textbooks, Cambridge University Press, 2021.