Swarms of Artificial Platelets for Emergent Hole Detection and Healing in Wireless Sensor Networks

Giada Simionato
giada.simionato@phd.unipi.it
Universities of Florence and Pisa
Florence and Pisa, Italy

Federico A. Galatolo
federico.galatolo@ing.unipi.it
University of Pisa
Pisa, Italy

Mario G.C.A. Cimino
mario.cimino@unipi.it
University of Pisa
Pisa, Italy

ABSTRACT

Most of the applications of wireless sensor networks require the continuous coverage of a region of interest. The irregular deployment of the nodes, or their failure, could result in holes in the coverage, thus jeopardizing such requirement. Methods to recover the sensing capabilities usually demand the availability of redundant full-fledged nodes, whose relocation should *heal* the holes. These solutions, however, do not consider the high cost of obtaining redundant, typically complex, devices, nor that they could in turn fail. In this work, we propose a bio-inspired and emergent approach toward hole detection and healing using a swarm of resource-constrained agents with reduced sensing capabilities, whose behavior draws inspiration from the concepts underlying blood coagulation. The swarm follows three rules: *activation*, *adhesion*, and *cohesion*, adapted from the behavior exhibited by platelets during the *human* healing process. Relying only on local and relative information, the mobile agents can detect the holes border and place themselves in locally optimal positions to temporarily restore the service. To validate the algorithm, we have developed a distributed, multi-process simulator. Experimental results show that the proposed method efficiently detects and heals the holes, outperforming two state-of-the-art solutions. It also demonstrates good robustness and flexibility to agent failure.

CCS CONCEPTS

• **Computer systems organization** → *Robotic autonomy*; • **Computing methodologies** → **Distributed artificial intelligence**.

KEYWORDS

Hole Detection, Hole Healing, Artificial Platelets, Swarm Intelligence, Emergent Cooperation

ACM Reference Format:
Giada Simionato, Federico A. Galatolo, and Mario G.C.A. Cimino. 2023. Swarms of Artificial Platelets for Emergent Hole Detection and Healing in Wireless Sensor Networks. In *Genetic and Evolutionary Computation Conference (GECCO '23), July 15–19, 2023, Lisbon, Portugal.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3583131.3590468

GECCO '23, July 15–19, 2023, Lisbon, Portugal

ACM ISBN 979-8-4007-0119-1/23/07…$15.00
https://doi.org/10.1145/3583131.3590468

1 INTRODUCTION

Wireless Sensor Networks (WSNs) are widely adopted for sensing and monitoring physical characteristics of a specific Region of Interest (ROI). Their pervasive nature makes them particularly suitable for a wide variety of applications, such as military surveillance [28, 33], environmental monitoring [27, 32], wildlife protection [17, 26], tracking [4, 11] and agriculture and farming [16, 23, 34].

Guaranteeing full coverage over time is a crucial requirement in many WSNs. However, establishing and maintaining such coverage can be challenging. Suboptimal deployment of the network nodes can result in portions of the ROI left uncovered [2, 15]. Moreover, nodes can undergo unexpected failures, caused, e.g., by hardware malfunctioning or energy depletion [8, 26]. Whenever one or more nodes fail, the corresponding area of the ROI can no longer be sensed, thus creating undesired *holes* in the network coverage. Most of the existing approaches to this problem assume that: (i) the WSN nodes are *mobile*, and can be relocated on-the-fly to restore the coverage (*heal* the holes); (ii) the WSN includes redundant nodes *from the start*, so that the relocation can take place without leaving other areas uncovered [13, 14, 27]. Such assumptions are unrealistic, as WSNs typically include static nodes, and adding redundant devices significantly increases the initial deployment cost. For these reasons, recovering from node failures in a short amount of time, with limited resources, is a challenging open problem.

In this work, we address the problem of hole detection and healing in time-sensitive and mission-critical scenarios, where a prompt response is fundamental but no nodes are preemptively deployed. We propose a novel swarm intelligence algorithm to temporarily restore the coverage using a swarm of resource-constrained agents with reduced sensing capabilities. Our approach is inspired by a parallelism between network coverage and human tissue, where holes can be thought of as wounds. The proposed emergent algorithm is based on the concepts underlying blood coagulation (Figure 1). To the best of our knowledge, this is the first study that presents a swarm intelligence algorithm based on the behavior of biological platelets in the context of WSNs. The agents are modeled as artificial platelets, that are initially released from one or more *release points* around the ROI. The swarm behaves according to three rules: (i) *activation*: upon detection of holes in the coverage, the agents activate, initiating the healing process. This strictly mirrors the behavior of blood platelets that activate and initiate coagulation after perceiving the injury; (ii) *adhesion*: when the agents reach the holes border, they start adhering to it, attracting other active agents. Similarly, the platelets adhere to the border of the injury and start secreting attractive substances; (iii) *cohesion*: active agents cohere to already deployed agents until the holes are completely

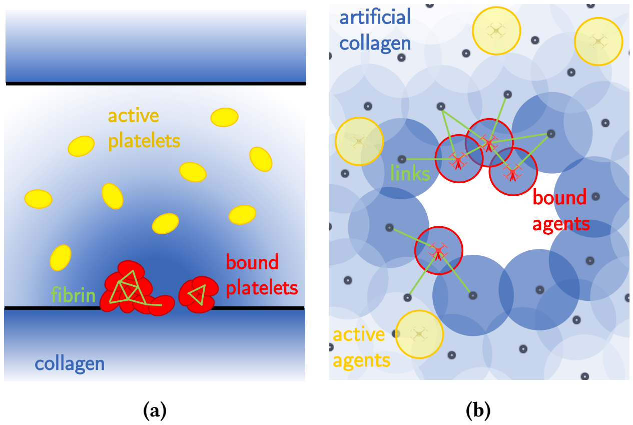


Figure 1: Visual parallelism between (a) the simplified process of blood coagulation and (b) the hole healing process in WSNs. Colors are used to better convey the parallelisms.

healed, thus restoring the coverage. This mimics the formation of the platelet plug responsible for healing the wounds.

To evaluate the performance of our algorithm, we developed a simulator for hole detection and healing problems. We performed several experiments under challenging scenarios, including noisy sensors, and the presence of multiple holes. Our results show the flexibility and robustness of the approach, which can quickly restore coverage even when the agents themselves are subject to failure. We stress that this last aspect has not been considered by previous works. Experimental studies show that the proposed method also outperforms two state-of-the-art solutions, namely [13] and [33].

The remainder of this paper is organized as follows. In Section 2, the literature is reviewed, while in Section 3 the behavior of biological platelets is introduced along with the context and notation. The proposed approach is thoroughly presented in Section 4, while in Section 5 experimental results are outlined and discussed. Finally, in Section 6 conclusions are drawn.

2 RELATED WORK

Hole Detection and Healing. Several studies have proposed ways to address hole detection [17, 18] or healing [2, 15] in WSNs. Works [14, 23] proposed two algorithms that minimize the energy consumption of mobile nodes, using a distributed Voronoi-based cooperation scheme [23] and insights from the intersection points among nodes [14]. However, both approaches only work with *dense* networks (i.e., way more nodes are deployed than what is needed to guarantee the coverage), a requirement that is usually not met. In [4], an additively weighted Voronoi diagram is used to detect the holes and to compute the patching positions, while [28] resorts to the Delaunay triangulation of the network combined with a virtual edge-based method. Both solutions are centralized, therefore lacking robustness, as opposed to our decentralized approach. Tree-based and chord-based hole and healing algorithms are proposed in [18, 26] as an alternative to Voronoi-based approaches. However, they require *homogeneous* networks, i.e., all nodes must have equal sensing capabilities. In [24] instead, a complex messaging protocol is used to detect the hole, which is then healed by driving nodes toward it using virtual forces. However, healing performance degrades in presence of multiple holes. To combine the advantages of geometry-based and virtual forces-based approaches, *hybrid* solutions were introduced [9, 16, 27], which however rely on the availability of GPS. Finally, [20, 32] use Particle Swarm Optimization (PSO) or an enhanced version of PSO to compute new positions for the patching nodes. Both approaches assume *global knowledge*, as opposed to our solution that only relies on local information.

Next, we describe in more detail two state-of-the-art methods that are closely related to our approach, i.e., [13, 33], although requiring several restrictive assumptions. Khalifa *et al.* [13] introduces DHDR, a solution that assumes a *dense* network of *mobile* nodes. The nodes move toward the hole when alerted by the missing reception of the heartbeat message from one of their neighbors. Therefore, *only the nodes adjacent to the hole* take part in the healing process. The displacement is computed using geometric criteria, taking as reference the intersection points between the sensing range of each node and that of its neighbors, following a heuristic priority scheme. Conversely, Yan *et al.* [33] treats the problem as an optimization task, proposing the FSHR algorithm. This enhances the artificial fish swarm algorithm [22] introducing the *leap* and *rebirth* behaviors. FSHR works with a set of inactive, mobile nodes scattered among a network of static nodes, relocating some of them to heal the holes. The WSN sink nodes compute the target positions and provide control inputs to guide the closest mobile nodes there. To do so, it discretizes the ROI into a grid and optimizes a fitness function given by the fraction of grid points covered by the union of the static nodes and the activated mobile nodes.

The abovementioned approaches share important limitations: *i)* they rely on GPS localization, which may not be available in mission-critical scenarios, whereas our method uses only local information; *ii)* they assume the availability of mobile, redundant nodes *in the original network* that can be relocated at need, while we use more practical external agents, independent of the network characteristics; and *iii)* they preemptively compute the new positions, without considering unexpected impairments that could prevent nodes from reaching their final locations. This type of offline planning usually fails when the environment is highly dynamic. Our method instead performs *online planning*, allowing the agents to dynamically adapt to the changes in the environment. We used [13] and [33] to benchmark our solution (Section 5.2.4).

Artificial Platelets. There exists a limited body of literature using the concepts behind biological platelets to model the behavior of robot swarms. The authors of [10] propose an algorithm based on PSO to control a swarm of artificial platelets to repair wounds in a simulated human vessel. [3] describes a control mechanism for nanorobots searching for wounds within the human body based on colored perceptive PSO. [21], instead, presents a theoretical approach to repair structures in inhospitable environments by means of a swarm of active particles. None of these works, however, deals with the problem of hole detection and healing in WSNs.

3 PRELIMINARIES

In this section we briefly introduce the concepts underlying blood coagulation, the context, and notation used throughout this paper.

3.1 Biological Platelets

Primary hemostasis is the process whereby a blood clot is formed to prevent the excessive loss of blood caused by vascular injuries [19]. The clot is made of platelets, that circulate in the human vasculature in *inactive* state. Upon vascular injury, the platelets become *activated* by the contact with collagen, a substance normally present in the extravascular space, but no longer isolated from the blood stream by the integrity of the endothelium [19]. Active platelets *adhere* to the injury, where the concentration of the collagen is highest. Upon adhesion, they start recruiting other platelets by emitting attractive substances, such as the adenosine diphosphate (ADP). The *bound* platelets, i.e., those adhered to the injury, provide surface for other platelets to *cohere* as to increase the size of the clot [19]. Bound platelets secrete also fibrins that bind them to stabilize the plug. By contrast, intact parts of the vessel produce prostacyclin that repulses the platelets, preventing the unnecessary and dangerous formation of clots [10]. Finally, to disband the platelets plug, the clot activates the plasmin that dissolves the bindings, forcing the platelets to return flowing in the vessels. For a more thorough explanation of the coagulation process, please refer to [1, 19].

3.2 Context Definition

In this work, we consider 2D obstacle-free environments where lies a network of fixed nodes with communication and sensing capabilities. The nodes are randomly deployed in the region of interest, ensuring full coverage while keeping limited the density of the network. The nodes can be treated as point-like entities, whose location coincides with the center of mass (CoM) of the device embodying it. For this reason, the network can be seen as a graph where a subset of nodes fails, generating holes in the service. To restore the coverage and heal the holes, we employ a swarm $\mathcal{S} = \{s_1, \cdots, s_n\}$ of moving *agents*, mounted on autonomous robots, e.g., drones, with limited resources and restricted sensing capabilities. To model the agents we resorted to a non-holonomic point-like representation, in which the agent pose coincides with that of the robot CoM. This abstraction provides broad applicability to different robots, which more complex dynamics can be eventually mapped back to the one we used to model the movement. The variables used as control inputs are the *steering* and *driving* speed. We will refer to each node or agent in the environment as *element*. Each element has two ranges, whose extent depend on whether it is a node or an agent. The *communication* range r_c is the maximum distance to which an element can exchange packets with other elements. The *sensing* range r_s is the maximum distance covered by the element. To model the communication and sensing capabilities we resort to the *Boolean disk* coverage model, that is a disk of range r_c, or r_s, centered in the CoM of each element. Inside the disk we assume perfect communication or sensing, while no communication or coverage can take place outside [31]. Since we consider agents with limited coverage capabilities, we suppose that their sensing radius $r_{s,a}$ is much less than the one of the nodes $r_{s,n}$, i.e., $r_{s,n} \gg r_{s,a}$. We assume that the range of communication is at least twice the sensing radius, i.e., $r_{c,a} \geq 2r_{s,a}$ and $r_{c,n} \geq 2r_{s,n}$ for agents and nodes, respectively. In this way, the connectivity among the elements in the restored area is guaranteed [35].

Table 1: Parallel between the biological and the swarm space.

Biological Space	*Swarm Space*
First hemostasis	Hole detection and temporary healing
Inactive/Active/Bound states	Inactive/Active/Bound states
Collagen for pl. activation	Positive ζ for agents activation
Concentration of collagen	Inverse of the ℓ values
Release of ADP to recruit	Bound agents updating ℓ to recruit
Prostacyclin to repel platelets	Considering only free virtual solutions
Release of fibrin for binding	Establishing links with the network

Apart from the sensor enabling the sensing in the chosen scenario, each element is equipped with a *range and bearing* (RaB) sensor to perceive the environment [6]. For an element e_i, the RaB provides, in its reference frame, the relative distance d_{ij} and angle φ_{ij} to all the other elements e_j within its communication range.

Throughout this paper, we will refer as *adjacent*, two elements with overlapping sensing disks. The *neighbors set* $\mathcal{N}_{e_i}$ of an element e_i is the set of all the elements adjacent to it.

4 ALGORITHM DESIGN

The proposed method comprises two stages: *hole detection* and *hole healing*. During the former, the swarm is initially resting in randomly scattered *release points* (RPs) in *inactive* state, mimicking the inactivity of the platelets normally flowing in the vessels. Mirroring the biological counterpart, the swarm is *activated* by the exposure to artificial collagen (AC), a virtual information forwarded over the network. For the healing stage, the *active* agents move toward the border of the hole, thanks to the information gathered through their RaB sensors. The resulting movement follows, therefore, the direction that can be interpreted as the gradient of the concentration of the artificial collagen: the higher the concentration of the AC, the closer they are to the hole. This strictly reminds biology, where a higher concentration of collagen means proximity to the wound. When the agents reach the frontier, they compute and move toward a locally optimal position, based on the information retrieved from the RaB sensor and on geometric criteria. Inspired by the repulsive action of prostacyclin, that prevents the platelets from adhering outside the wound, the target locations for the agents deployment cannot lie in areas already covered by either nodes or agents (outside the hole). Upon deployment, the agents switch to the *bound* state and establish connections with the rest of the network, mimiking the effects of the release of fibrin as plug stabilizer. The bound agents update their AC information, to reflect the changes in the hole border. This results in the agents being part of the recruitment process of other agents, as happens biologically with the release of the attractive ADP by bound platelets. The temporary network composed by a sufficient number of bound agents, thus fully covering the desired area, will remain in place until the damages are fixed. Upon the restoration of the faulty nodes, the swarm returns to its RPs or flies to cover new holes, as happen when the tissue heals and the clot dissolves in the blood (see Section 3.1). We will refer to nodes of the network or bound agents as *deployed elements*. Table 1 reports the parallelisms between the biological and swarm space, while the pseudocode of the proposed method is reported in Algorithm 1.

4.1 Hole Detection

In order to detect the presence of holes in the network, each node and bound agent periodically computes the *Angular Coverage Ratio* (ACR) $c \in [0, 1]$, a coefficient representing the fraction of the boundary of its sensing area covered by its deployed neighbors. The ACR of a deployed element o_i, i.e., c_i, is computed using information provided by its RaB sensor, namely $(d_{ij}, \varphi_{ij}) \ \forall o_j \in \mathcal{N}_{o_i}$. At first, all the angular spans corresponding to the arcs covered by the neighbors are computed according to Equation (1a), where r_{s,o_i} is the sensing range of the deployed element o_i and r_{s,o_j} of o_j. The angular spans are then centered in the respective relative bearing φ_{ij} and the resulting intervals are merged as in Equation (1b). Finally, c_i is obtained as in Equation (1c). Figure 2 shows the geometric relations among the quantities involved. If the ratio c_i is below a certain threshold Th_c, the deployed element o_i is on the boundary of a hole, i.e., $o_i \in \mathcal{B}$.

$$\theta_j = \arccos\left(\frac{r_{s,o_i}^2 + d_{ij}^2 - r_{s,o_j}^2}{2 r_{s,o_i} d_{ij}}\right) \tag{1a}$$

$$\Theta_i = \bigcup_{o_j \in \mathcal{N}_{o_i}} [\varphi_{ij} - \theta_j, \varphi_{ij} + \theta_j] \tag{1b}$$

$$c_i = |\Theta_i| / 2\pi \tag{1c}$$

Each deployed element retains a Boolean value ζ representing whether at least one hole is present in the network. This information is spread in the network originating from the deployed elements in $\mathcal{B}$, through a custom version of a *gossip protocol* [12]. A positive value of ζ represents the presence of artificial collagen that, once spread, reaches the RPs causing the agents to switch their state from *inactive* to *active* (Alg. 1, lines 1-4).

Each deployed element o_i is also associated with a *level* ℓ_i, an integer value representing the hop-distance to the boundary of the nearest hole and computed according to Equation (2). This can be seen as the inverse of the concentration of the artificial collagen.

$$\ell_i = \begin{cases} 0, & \text{if } o_i \in \mathcal{B} \\ \min\limits_{o_j \in \mathcal{N}_{o_i}} \ell_j + 1, & \text{otherwise} \end{cases} \tag{2}$$

The levels give rise to a form of potential field in the network, that the swarm can follow to reach the nearest hole. Each deployed element has access to the type, the state (if agent), the level ℓ, the value ζ, the distance and bearing of all the elements within its communication range. These quantities can be retrieved from the range and bearing sensor or through wireless communication.

4.2 Hole Healing

Upon activation, the swarm leaves the release points and uses the information provided by the RaB sensors to follow the inverse of the gradient of the potential field generated by the levels of the deployed elements. To do so, the agents move toward the nearest deployed element with minimum level among those perceived (Alg. 1, line 15, 17). With the deployed elements at the boundary of the holes having global minimum level, the swarm will eventually reach a hole. They move in the computed direction imposing adaptive control values to their actuators. Unless different values of the controls are explicitly required to accomplish maneuvers or to approach a destination point, the agents move with a constant steering $\bar{\omega}$ and driving $\bar{v}$ speed.

Algorithm 1: Agent Logic

```
1  $state \leftarrow inactive$;
2  if $broken \in gossip\_messages()$ then
3      $state \leftarrow active$;
4  end
5  while $state$ is $active$ do
6      $RaB \leftarrow range\_and\_bearing()$;
7      $nodes \leftarrow o_i \in RaB \mid \ell_i = 0$ and $o_i$ is $node$;
8      $agents \leftarrow o_i \in RaB \mid \ell_i = 0$ and $o_i$ is $agent$;
9      $pn, pa, pm, pd \leftarrow select\_parents(nodes, agents)$;
10     $proposals \leftarrow compute\_solutions(pn, pa, pm, pd)$;
11     $solutions \leftarrow filter\_free(proposals)$;
12     if $solutions \neq \emptyset$ then
13         $target \leftarrow rank\_solutions(solutions)$;
14     else
15         $target \leftarrow select\_min\_element(RaB)$;
16     end
17     $drive\_to(target)$;
18     if $d(agent, target) \leq Th_b$ then
19         $state \leftarrow bound$;
20     end
21 end
```

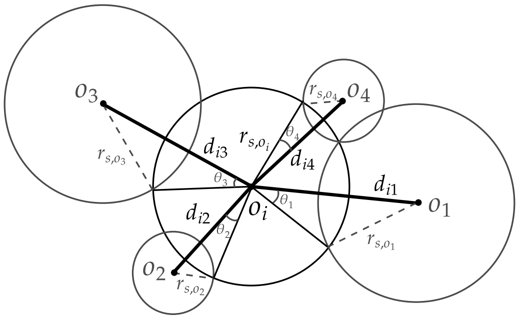


Figure 2: Geometric interpretation of the ACR.

When an agent perceives at least two *adjacent* deployed elements with level 0 (named *parents*), it geometrically computes a list of virtual targets that represent its locally optimal deployment positions. Specifically, it creates a list of parents from the zero-level deployed elements perceived using its RaB sensor (line 9). To determine the adjacency of two deployed elements, their distance is obtained through triangulation techniques. For each pair of parents, the agent computes two solutions (line 10). In case of two parents of the same type (two nodes or two bound agents), the locally optimal positions are reported in Figure 3a. We consider $\mathbf{o_i}$ and $\mathbf{o_j}$ as the vectors of Cartesian coordinates of the two parents o_i and o_j, respectively, in the reference frame of the agent. To determine the two solutions, we compute the displacements d_b along the unit vector $\mathbf{k}$ joining the CoMs of the two parents, and d_h along the unit vector perpendicular to $\mathbf{k}$, i.e., $\mathbf{k}_\perp$, as in Equation (3). The two solutions are then computed according to Equation (4).

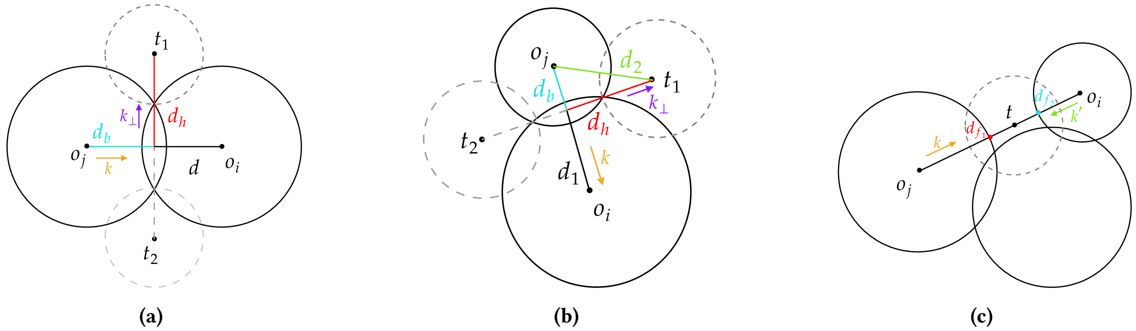


Figure 3: Locally optimal virtual targets in case of parents with (a) same or (b) different sensing radius. (c) Additional target.

$$\mathbf{k} = \frac{\mathbf{o_i}-\mathbf{o_j}}{\|\mathbf{o_i}-\mathbf{o_j}\|} = \frac{\mathbf{o_i}-\mathbf{o_j}}{d} \qquad d_b = d/2 \qquad d_h = \sqrt{r_{s,o_j}^2 - \frac{d^2}{4}} + r_{s,a} \tag{3}$$

$$\mathbf{t}_{1,2} = \mathbf{o_j} + d_b\mathbf{k} \pm d_h\mathbf{k}_\perp \tag{4}$$

For two parents of different type, the locally optimal solutions are shown in Figure 3b and computed as in Equation (5) and (4).

$$\mathbf{k} = \frac{\mathbf{o_i}-\mathbf{o_j}}{\|\mathbf{o_i}-\mathbf{o_j}\|} = \frac{\mathbf{o_i}-\mathbf{o_j}}{d_1} \qquad d_2 = 2\sqrt{r_{s,a}^2 - \left(\frac{d_1^2-r_{s,n}^2-r_{s,a}^2}{2r_{s,n}}\right)^2}$$
$$p = \frac{2d_1+d_2}{2} \qquad A = \sqrt{p(p-d_1)^2(p-d_2)} \tag{5}$$
$$d_h = 2A/d_1 \qquad d_b = \sqrt{d_2^2 - d_h^2}$$

To improve the coverage, we considered additional virtual solutions, originating from pairs of non-adjacent deployed elements with level zero but with distance $d_{ij} \leq r_{s,o_i} + r_{s,o_j} + 2r_{s,a}$. For each pair of such parents, the positioning point is the midpoint of the segment joining the two intersections of the boundaries of the sensing disks with the segment joining the CoMs of the parents, as depicted in Figure 3c. Equation (6) shows the computation of the additional solution $\mathbf{t}$.

$$\mathbf{k} = \frac{\mathbf{o_i}-\mathbf{o_j}}{\|\mathbf{o_i}-\mathbf{o_j}\|} \qquad \mathbf{k}' = \frac{\mathbf{o_j}-\mathbf{o_i}}{\|\mathbf{o_i}-\mathbf{o_j}\|}$$
$$\mathbf{d_{f_1}} = \mathbf{o_j} + r_{s,o_j}\mathbf{k} \qquad \mathbf{d_{f_2}} = \mathbf{o_i} + r_{s,o_i}\mathbf{k}' \qquad \mathbf{t} = (\mathbf{d_{f_1}} + \mathbf{d_{f_2}})/2 \tag{6}$$

The list of solutions is then filtered (line 11), removing all those lying within the sensing disk of a perceived deployed element. This results in keeping only the *free* solutions, i.e., all the solutions $\mathbf{t}$ such that $d(\mathbf{t}, \mathbf{o_j}) > r_{s,o_j}\ \forall o_j \in O$, where O is the set of all the deployed elements perceived by the agent. The free solutions are ranked (line 13) according to a *ranking policy* based on the distance and type of parents. Each agent moves toward the highest ranking solution in the list, until either the selected target is not free anymore or its distance is lower than a threshold Th_b (line 18). In the former case, the agent repeats the process and selects a new target, while in the latter, it stops and deploys itself in the network, establishing connections with its neighbors and switching its state to *bound* (line 19). This deployment restores the service in the area covered by the agent. When part of the network, the agent starts computing the ACR as in Equation (1c), thus updating its level ℓ and contributing to the gossip protocol. The deployment of the agents in the network, therefore, changes the shape of the boundary of the holes and modifies the potential field. This attracts other agents, mirroring the ADP release in the biological counterpart.

5 PERFORMANCE EVALUATION

In this section, we present an intensive evaluation of the performance of the proposed algorithm.

5.1 Experimental Setup

To perform the experiments we have developed a discrete-time simulator, whose architecture was purposely designed for handling multi-agent applications with a high level of scalability. The simulator and our approach was implemented in Python and made available in [25] to foster reproducibility.

In the experiments we considered a network of 125 randomly placed nodes. For the communication and sensing capabilities of nodes and agents, we used quantities that meet the assumptions of Section 3.2 and are compliant with existing technologies, see, e.g., [29]. The control speeds of the agents were set to comparable values to those of commercial quadrotors [5]. We modeled the error affecting the RaB sensors with two Gaussian noises having fixed variance (bearing) and variance proportional to the measured distance (range). Specifically, by default, we set the distance noise standard deviation to 1.5% of the measurement value, and the angle standard deviation to 5°. However, to evaluate the robustness of the proposed method we also experimented with different noise levels as detailed in Section 5.2.3 and Table 2.

For each experiment we performed 100 simulations lasting $T = 1000$ time steps each. We set the duration of a time step to 1 s. As a default configuration, we considered a single release point of 50 agents to heal a single hole caused by the failure of 7 adjacent nodes. The simulation parameters are reported in Table 2.

The layout of the network, the position of the RPs, the shape and position of the holes were randomized for each simulation. We stress that using a single release point represents a more realistic situation in emergency scenarios, where it is usually difficult to access the ROI. This also represents the most challenging configuration for evaluating our approach, as further discussed in Section 5.2.2.

To evaluate the performance, we analyzed the trend of the average coverage over time. That is, the average fraction of the area left uncovered by failed nodes that was restored by the agents. In

Table 2: Parameters used in the simulations. The default values are highlighted in bold font.

Parameter	*Symbol*	*Value*
Node sensing range	$r_{s,n}$	30 m
Node communication range	$r_{c,n}$	60 m
Agent sensing range	$r_{s,a}$	10 m
Agent communication range	$r_{c,a}$	60 m
Agent cruise driving speed	$\bar{v}$	5 m/s
Agent cruise steering speed	$\bar{\omega}$	0.1 rad/s
Avg. Perc. RaB distance error	ρ	{0, 1, **1.5**, 5, 20} %
Avg. error on RaB bearing	γ	{0, 2, **5**, 15, 20} °
Bound threshold	Th_b	0.3 m
Coverage threshold	Th_c	{0.85, 0.90, **0.95**, 1.0}
Avg. Perc. agent failure	P_f	{**0**, 5, 10, 25} %

addition, we considered the average number of agents required to heal the holes. We will refer as *convergence*, the attainment of a stable value by the coverage before the simulation ends.

5.2 Results and discussion

In the following, we report the experiments and results that were carried out to evaluate different aspects of the proposed approach. A visual representation of the healing process in case of multiple holes in the network is provided in Figure 4.

5.2.1 Parameters Variation. To evaluate our approach, we analyzed the impact of the variation of the two parameters that yielded the most variance in the performance [7]: the *ranking policy* and Th_c, introduced in Section 4.1.

Regarding the former, we carried out sets of simulations with the policies *DNAM, NAMD, AMND, closest* and *random.* The name of the first three strategies reminds the descending priority given to the computed virtual solutions according to the type of their parents, i.e., Nodes, Agents, Mixed and aDditional solutions. Among solutions of the same type, priority is given to the closest one. *closest* always selects the closest computed solution, while *random* chooses the closest solution among those of a randomly selected type of parents. We used the default configuration and the parameters in Table 2. Figure 5a shows the trends of the average coverage over time. We observe that *AMND* and *NAMD* do not reach the same level of coverage as the other policies, due to the fact that some agents keep moving between the same solutions (of higher priority). This is caused by the limited range of the RaB sensor, not always allowing the agent to perceive that the chosen solution is not *free* until it gets sufficiently close. The *random* strategy, randomly combining all the approaches, and *DNAM* provide high coverage with slightly less deployed agents on average. However, they show longer transient to convergence, therefore not representing the best strategies for time sensitive scenarios. Overall, the *closest* policy achieves the best results, therefore we selected it for our experiments.

Figure 5b, instead, shows the trends of the average coverage over time for the values of Th_c reported in Table 2. As for the previous experiments, we have relied on the default configuration. We recall from Section 4.1 that Th_c is the threshold for the ACR c_i under which a node or a bound agent o_i knows it is on the border of a hole. For low values of Th_c, e.g., 0.85, most of the deployed elements keep a level ℓ different from 0, therefore not offering enough surface for the active agents to adhere. Conversely, for values of Th_c too high, such as 1.0, the majority of bound agents set their level to 0, representing ubiquitous attractive signals that could overwhelm the active agents. The best performance is obtained with $Th_c = 0.95$, so we kept this value for the rest of the evaluation process.

Table 3: Results of multi-hole and multi-RPs experiments.

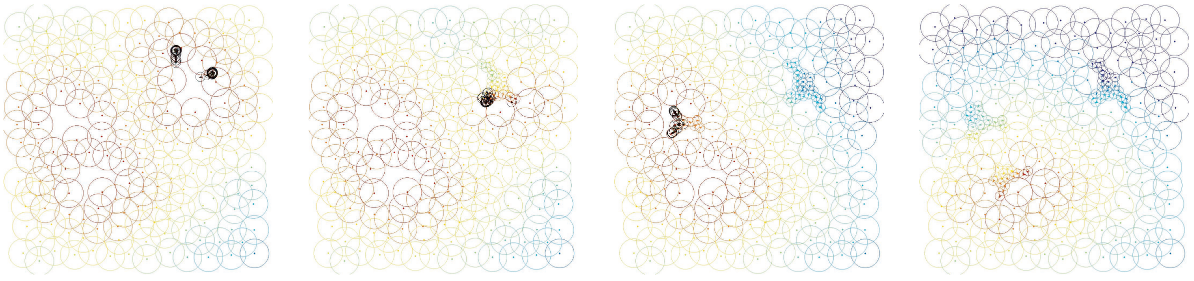
	3 RPs		**5 RPs**	
	Avg.	95% CI	Avg.	95% CI
Coverage	81.56%	2.96%	87.96%	2.46%
Bound agents	34.84	1.49	36.34	1.34

5.2.2 Scalability and Flexibility Analysis. To capture the scalability of our approach we firstly studied the variations in the coverage trends for different sizes of the hole. Specifically, we varied m, the number of adjacent nodes that, failing, generate the hole. For $m = \{1, 4, 7, 10, 15\}$, the results are shown in Figure 5c. To ensure a fair comparison among experiments, we increased the number of agents in the RP accordingly (i.e., $n = \{5, 25, 50, 75, 125\}$). In this way, the possible lack of convergence in the coverage cannot be caused by the scarcity of agents. As expected, the larger the hole, the longer the transient will be, since incrementally bigger areas need to be covered. Our method appears to be quite scalable for holes of different dimensions, even in presence of massive failures (e.g., $m = 15$), since it usually converges to values greater than 96%. However, in case of a single failure, we observe a reduction of the coverage at convergence. This is caused by the restricted pool of available positions where to deploy the agents. In this case, the area left uncovered is usually so small that spending an agent to cover it would be counterproductive.

Concerning the flexibility, we performed sets of simulations with the number of release points varying in $\{1, 3, 5\}$ but keeping the total size of the swarm constant to 50 for a fair comparison. We used the default configuration of Table 2. As expected, with multiple RPs we observed a speed up to the convergence, shown in Figure 5d. The shorter transient is explained by the fact that, on average, the probability of having a release point closer to the hole is higher. Moreover, different RPs allow approaching the hole from different directions, accelerating the healing process.

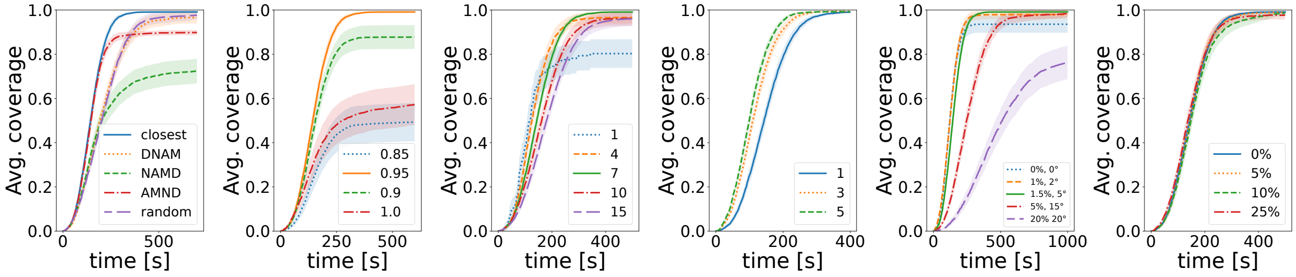
Finally, to further prove the flexibility of the algorithm, we carried out experiments with *multiple* holes and multiple release points (i.e., 3 and 5). Specifically, we considered three holes caused by the failure of groups of 3, 3, and 4 adjacent nodes. As in the previous cases, we kept constant the size of the swarm, to ensure fairness in the comparison. When compared to Figure 5d, the results reported in Table 3 show that our algorithm still provides good coverage in nearly the same time required for single hole scenarios.

5.2.3 Robustness Analysis. To evaluate the robustness of our algorithm, we performed a set of experiments varying the standard deviation of the errors affecting the RaB measurements, as summarized in Table 2. These values are comparable to the errors that typically affect sensors of different quality. We also included the case of ideal ($\rho = 0$, $\gamma = 0$) and heavily perturbed ($\rho = 20\%$, $\gamma = 20°$)



(a) (b) (c) (d)

Figure 4: Snapshots of a multi-hole healing process for subsequent time steps. Agents are represented as triangles. Nodes are depicted as points. Active agents are colored in black, while nodes and bound agents follow the palette that visually conveys the gradient formed by their level ℓ. Only sensing ranges are shown for clarity. (a) t = 10 s; (b) t = 472 s; (c) t = 634 s; (d) t = 913 s.



(a) Ranking policy. (b) Th_c. (c) Hole dimension. (d) Multiple RPs. (e) RaB errors. (f) Agents failure.

Figure 5: Average coverage over time. The shaded bands represent the 95% confidence interval.

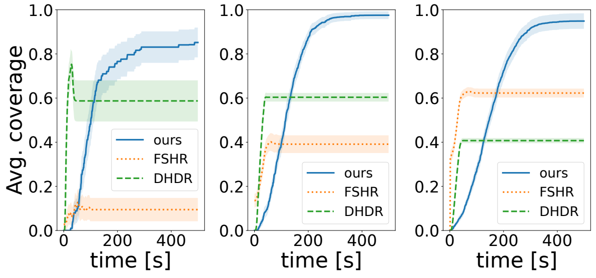
perception. Figure 5e shows the coverage trends for these experiments, presenting slower convergence with increasing severity of the error. The reason is that a higher error introduces higher variation between consequent perceptions. This results in the agent frequently changing its target deployment position, and thus direction of movement. This, in turn, will end up in a swinging behavior that can stall the agent, slowing down the healing process. Interestingly, in the case of ideal perception, we observe slightly worse performance. This happens because the optimal positions selected by the agents are only *locally* optimal, and therefore might not coincide with the globally optimal ones. For this reason, the noise introduced by the sensor may allow the agents to escape from points of local maximum coverage that are globally suboptimal. Overall, our method is quite robust to different levels of perception error: precise localization is only needed at short distances (where the error is usually limited), while on longer distances a rough indication about the direction is sufficient.

One of the most promising features of our algorithm is its ability to recover from agent failure. To support our claim we carried out experiments in which, on average, a fraction P_f of agents will independently fail during each simulation. To do so, we make each agent fail with probability $p = 1 - (1 - P_f)^{1/T}$ at every step of the simulation. In Figure 5f we show the coverage trends for $P_f = \{0, 0.05, 0.1, 0.25\}$. In this evaluation, we used the default parameter setting of Table 2. For a fair comparison among different values of P_f, as described in Section 5.2.2, we increased the size of the swarm by exactly the average amount of agents that will fail in that set of simulations. Figure 5f shows no statistically significant difference in the trends, meaning that our algorithm is robust enough to cope with the loss of agents during the healing process. This is achieved because there is no pre-defined assignment of the target positions to the agents: if one fails, another one can carry out the healing process. If a bound agent fails, other agents that are currently healing the hole are close enough to promptly take its place.

Table 4: Comparison for single-hole single-failure experiments, with (w/) and without (w/o) noisy perception.

	DHDR [13]		FSHR [33]		ours	
	Avg. %	95% CI	Avg. %	95% CI	Avg. %	95% CI
w/o noise	57.19	11.27	17.84	6.62	**76.17**	7.86
w/ noise	58.71	9.37	9.60	5.16	**88.08**	5.87

5.2.4 State-of-the-art comparison. We compared our approach with two recent hole detection and healing algorithms: DHDR [13] and FSHR [33] (see Section 2). To ensure a fair comparison in terms of the duration of the healing process, we endowed the two methods with the same dynamics of movement of our approach, without affecting their logic. We evaluated all methods using nodes and



(a) $m = 1$. **(b)** $m = 7$. **(c)** $m = 15$.

Figure 6: Average coverage over time for different hole size. The shaded bands represent the 95% confidence interval.

agents with equal sensing range, as to comply with DHDR requirements. For each set of experiments, of 130 runs each, we tested the three methods on the same set of randomly generated scenarios. We deployed the same number of agents (ours), redundant nodes (DHDR), and mobile nodes (FSHR), to guarantee equal healing capacity. We used the default parameters in Table 2 for our method and those reported in [13] for DHDR and [33] for FSHR. We recall that, for all methods, the coverage is computed as the ratio between the recovered area over the initial hole area, as opposed to [13, 33], where the reference area is the whole ROI. In this way, the performance is not misled by the massive contribution brought by the intact network.

We performed 130 simulations in the case of a single hole caused by a single node failure, as in [13, 33]. We conducted the experiments in case of ideal and noisy perception, deploying 4 agents for each method. Results are reported in Table 4. We perturbed the GPS of DHDR and FSHR using a zero mean Gaussian model with st.dev. of 4.9m [30]. Table 4 shows that our method, as previously discussed, performs slightly better with noisy perceptions. Conversely to DHDR, FSHR strongly relies on precise positioning information, therefore with noisy values it yields worse performance.

Keeping the perception perturbed, we carried out experiments varying the number of failed nodes causing the hole in $m = \{7, 15\}$. For all methods, we increased the number of agents to 10 and 25, respectively, as to guarantee sufficient healing capacity. The results are reported in Figure 6. Our method always achieve high coverage, especially when the size of the hole is bigger (Section 5.2.2). DHDR for $m = 1$ (Figure 6a) shows a peak in the coverage representing its potential healing capacity. The decrease in the coverage is caused by the undue movement of the nodes chosen for recovery: their final positions cross the optimal points (corresponding to the peak), leaving other parts uncovered. This method seems to reach similar level of coverage at convergence in case of $m = 1$ and $m = 7$, however the absence of the peak in the latter case (Figure 6b) suggests that the attained coverage is the maximum reachable by this method and not caused by misplacement. The trend is corroborated by Figure 6c, where for bigger holes DHDR reaches lower coverage. This is caused by the involvement of only the nodes at the hole border for healing: they can only move by a certain extent to avoid leaving other parts uncovered. Instead, FSHR achieves better performance when increasing m: the bigger the hole the more grid points of the discretized ROI can be used to guide the optimization problem, resulting in better positions for the mobile nodes. Overall, our approach shows higher coverage than DHDR and FSHR, but slower transient to convergence. DHDR and FSHR make use of the closest nodes for healing (on the border or scattered in the area), while our method releases the agents from farther RPs.

Table 5: Results of the comparison for single-hole 7-failure experiments, for different P_f values.

	DHDR [13]		FSHR [33]		ours	
P_f	Avg. %	95% CI	Avg. %	95% CI	Avg. %	95% CI
0	60.38	1.96	38.80	3.76	**97.53**	1.62
0.05	33.48	4.62	34.77	3.69	**97.13**	1.76
0.1	8.29	6.34	34.23	3.65	**97.58**	1.63
0.25	5.10	6.80	30.90	3.64	**97.12**	1.59

Finally, we tested the three approaches for the robustness to agent failure, when healing a hole caused by 7 faulty nodes and relying on perturbed perception. We used the same setting of Section 5.2.3, with again $P_f = \{0.05, 0.1, 0.25\}$. As previously stated, to guarantee sufficient healing capacity, we increased the number of agents to 11, 12, 13, respectively. The results shown in Table 5 highlight the robustness of our method, while showing the severe drop in the coverage for DHDR and FSHR. This latter seems to be more resilient to agent failure. However, as suggested by the overall low coverage achieved, the failure mainly interests mobile nodes deployed on parts of the ROI already covered by static nodes. For this reason, their failure does not strongly impact the coverage.

The experiments showed that, on average, our method involves 56.3% less agents than DHDR and 36.23% less agents than FSHR.

6 CONCLUSIONS

In this work we addressed the problem of hole detection and healing in WSNs. We proposed a novel swarming approach inspired by the concepts underlying blood coagulation. The agents in the swarm, modeled as artificial platelets, use relative localization and information gathered from the network, to detect the holes and place in locally optimal positions, restoring the coverage. Experimental results demonstrate that our algorithm efficiently reinstates the service in uncovered areas, showing good flexibility and robustness. Future research directions point to the inclusion of collision avoidance capabilities and a model for the residual energy level of the agents. Finally, we will work toward the addition of obstacles in the space while exploring different positioning approaches.

ACKNOWLEDGMENTS

Work partially funded by: (i) the Italian Ministry of University and Research via the FoReLab project (Departments of Excellence), and via the National Recovery and Resilience Plan - National Center for Sustainable Mobility MOST/Spoke10; (ii) the Italian Ministry of Economic Development via the "Agreements for Innovation" Programme, "4D Drone Swarms" (4DDS) project; (iii) the University of Pisa, via the PRA_2022_101 project “Decision Support Systems for territorial networks for managing ecosystem services”.

REFERENCES

[1] Mohammad Feroz Alam and Khaliqur Rahman. 2022. Platelet substitutes. In *Nanotechnology for Hematology, Blood Transfusion, and Artificial Blood*. Elsevier, 429–449.

[2] Lynda Aliouane and Mahfoud Benchaiba. 2014. HACH: healing algorithm of coverage hole in a wireless sensor network. In *2014 Eighth international conference on next generation mobile apps, services and technologies*. IEEE, 215–220.

[3] Davide Ceraso and Giandomenico Spezzano. 2016. Controlling swarms of medical nanorobots using CPPSO on a GPU. In *2016 International Conference on High Performance Computing & Simulation (HPCS)*. IEEE, 58–65.

[4] Mansoor Davoodi, Esmaeil Delfaraz, Sajjad Ghobadi, and Mahtab Masoori. 2021. Hole Detection and Healing in Hybrid Sensor Networks. *arXiv preprint arXiv:2106.10659* (2021).

[5] DJI. 2023. https://www.dji.com/it/mini-2/specs

[6] Álvaro Gutiérrez, Alexandre Campo, Marco Dorigo, Jesus Donate, Félix Monasterio-Huelin, and Luis Magdalena. 2009. Open e-puck range & bearing miniaturized board for local communication in swarm robotics. In *2009 IEEE International Conference on Robotics and Automation*. IEEE, 3111–3116.

[7] Kyle Robert Harrison, Beatrice M Ombuki-Berman, and Andries P Engelbrecht. 2019. An analysis of control parameter importance in the particle swarm optimization algorithm. In *Advances in Swarm Intelligence: 10th International Conference, ICSI 2019, Chiang Mai, Thailand, July 26–30, 2019, Proceedings, Part I 10*. Springer, 93–105.

[8] D Jewel, P Brundha, DJW Wise, and G Aravind Swaminathan. 2016. Improved hole detection healing and replacing algorithm for optimal coverage in wireless sensor networks. *Int. J. Sci. Res. Sci. Eng. Tech* 2 (2016), 724–731.

[9] Renuka Kadu and Kalpana Malpe. 2017. Movement-assisted coverage improvement approach for hole healing in wireless sensor networks. In *2017 Second International Conference on Electrical, Computer and Communication Technologies (ICECCT)*. IEEE, 1–4.

[10] Boonserm Kaewkamnerdpong, Pinfa Boonrong, Supatchaya Trihirun, and Tiranee Achalakul. 2015. Modeling nanorobot control using swarm intelligence for blood vessel repair: A rigid-tube model. In *Adaptation and Hybridization in Computational Intelligence*. Springer, 205–236.

[11] Zhiping Kang, Honglin Yu, and Qingyu Xiong. 2013. Detection and recovery of coverage holes in wireless sensor networks. *Journal of Networks* 8, 4 (2013), 822.

[12] Anne-Marie Kermarrec and Maarten Van Steen. 2007. Gossiping in distributed systems. *ACM SIGOPS operating systems review* 41, 5 (2007), 2–7.

[13] Banafsj Khalifa, Zaher Al Aghbari, and Ahmed M Khedr. 2021. A distributed self-healing coverage hole detection and repair scheme for mobile wireless sensor networks. *Sustainable Computing: Informatics and Systems* 30 (2021), 100428.

[14] Ahmed M Khedr, Walid Osamy, and Ahmed Salim. 2018. Distributed coverage hole detection and recovery scheme for heterogeneous wireless sensor networks. *Computer Communications* 124 (2018), 61–75.

[15] Abdelkader Khelil, Rachid Beghdad, and Amar Khelloufi. 2020. 3HA: hybrid hole healing algorithm in a wireless sensor networks. *Wireless Personal Communications* 112, 1 (2020), 587–605.

[16] Nikitha Kukunuru, Davuluri RajyaLakshmi, and Avula Damodaram. 2014. Hybrid approach for detecting and healing the coverage-hole in Wireless Sensor Network. In *2014 International Conference on Signal Propagation and Computer Technology (ICSPCT 2014)*. IEEE, 110–115.

[17] Prasan Kumar Sahoo, Ming-Jer Chiang, and Shih-Lin Wu. 2016. An efficient distributed coverage hole detection protocol for wireless sensor networks. *Sensors* 16, 3 (2016), 386.

[18] Xiaoyun Li, David K Hunter, and Kun Yang. 2006. Wlc12-1: Distributed coordinate-free hole detection and recovery. In *IEEE Globecom 2006*. IEEE, 1–5.

[19] Kathryn G Link, Matthew G Sorrells, Nicholas A Danes, Keith B Neeves, Karin Leiderman, and Aaron L Fogelson. 2020. A mathematical model of platelet aggregation in an extravascular injury under flow. *Multiscale Modeling & Simulation* 18, 4 (2020), 1489–1524.

[20] Shalu Mehta and Amita Malik. 2020. A swarm intelligence based coverage hole healing approach for wireless sensor networks. *EAI Endorsed Transactions on Scalable Information Systems* 7, 26 (2020), e8–e8.

[21] John Oyekan. 2021. Multi-Objective Optimisation of Robotic Active Particle Swarms for Continuous Repair of Large Scale High Value Structures. In *2021 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 1312–1318.

[22] Farhad Pourpanah, Ran Wang, Chee Peng Lim, Xi-Zhao Wang, and Danial Yazdani. 2023. A review of artificial fish swarm algorithms: Recent advances and applications. *Artificial Intelligence Review* 56, 3 (2023), 1867–1903.

[23] Chenxi Qiu, Haiying Shen, and Kang Chen. 2017. An Energy-Efficient and Distributed Cooperation Mechanism for k-Coverage Hole Detection and Healing in WSNs. *IEEE Transactions on Mobile Computing* 17, 6 (2017), 1247–1259.

[24] Mustapha Reda Senouci, Abdelhamid Mellouk, and Khalid Assnoune. 2013. Localized movement-assisted sensordeployment algorithm for holedetection and healing. *IEEE Transactions on parallel and distributed systems* 25, 5 (2013), 1267–1277.

[25] Giada Simionato and Federico A. Galatolo. 2023. https://github.com/GiadaSimionato/HDHSim.git

[26] Parmod Singh and Yaw-Chung Chen. 2020. Sensing coverage hole identification and coverage hole healing methods for wireless sensor networks. *Wireless Networks* 26, 3 (2020), 2223–2239.

[27] Chakchai So-In, Tri Gia Nguyen, and Nhu Gia Nguyen. 2019. An efficient coverage hole-healing algorithm for area-coverage improvements in mobile sensor networks. *Peer-to-Peer Networking and Applications* 12, 3 (2019), 541–552.

[28] A Soundarya and V Santhi. 2017. An efficient algorithm for coverage hole detection and healing in wireless sensor networks. In *2017 1st International conference on electronics, materials engineering and nano-technology (IEMENTech)*. IEEE, 1–5.

[29] Terabee. 2021. https://terabee.b-cdn.net/wp-content/uploads/2021/02/Specification-Sheet-Evo-60m.pdf

[30] Frank Van Diggelen and Per Enge. 2015. The world's first GPS MOOC and worldwide laboratory using smartphones. In *Proceedings of the 28th international technical meeting of the satellite division of the institute of navigation (ION GNSS+ 2015)*. 361–369.

[31] Bang Wang. 2010. *Coverage control in sensor networks*. Springer Science & Business Media.

[32] Jin Wang, Chunwei Ju, Hye-jin Kim, R Simon Sherratt, and Sungyoung Lee. 2019. A mobile assisted coverage hole patching scheme based on particle swarm optimization for WSNs. *Cluster Computing* 22, 1 (2019), 1787–1795.

[33] Luoheng Yan, Yuyao He, and Zhongmin Huangfu. 2020. A fish swarm inspired holes recovery algorithm for wireless sensor networks. *International Journal of Wireless Information Networks* 27, 1 (2020), 89–101.

[34] Shuangjiao Zhai, Zhanyong Tang, Dajin Wang, Zhanglei Li, Xiaojiang Chen, Dingyi Fang, and Feng Chen. 2017. Coverage hole detection and recovery in wireless sensor networks based on RSSI-based localization. In *2017 IEEE international conference on computational science and engineering (CSE) and IEEE international conference on embedded and ubiquitous computing (EUC)*, Vol. 2. IEEE, 250–257.

[35] Honghai Zhang, Jennifer C Hou, et al. 2005. Maintaining sensing coverage and connectivity in large sensor networks. *Ad Hoc Sens. Wirel. Networks* 1, 1-2 (2005), 89–124.