A

Accurate, Dynamic, & Distributed Localization of Phenomena for Mobile Sensor Networks

CHRISTOS ANAGNOSTOPOULOS, School of Computing Science, University of Glasgow, G12 8QQ UK, e-mail: christos.anagnostopoulos@glasgow.ac.uk
STATHES HADJIEFTHYMIADES, Department of Informatics & Telecommunications, National and Kapodistrian University of Athens, 15784 Greece, e-mail: shadj@di.uoa.gr
KOSTAS KOLOMVATSOS, Department of Computer Science, University of Thessaly, 35100 Greece, e-mail: kolomvatsos@cs.uth.gr

We present a robust, dynamic scheme for the automatic self-deployment and relocation of mobile sensor nodes (e.g., unmanned ground vehicles, robots) around areas where phenomena take place. Our scheme aims (i) to sense environmental contextual parameters and accurately capture the spatio-temporal evolution of a certain phenomenon (e.g., fire, air contamination) and (ii) to fully automate the deployment process by letting nodes relocate, self-organize (and self-reorganize) and optimally cover the focus area. Our intention is to 'opportunistically' modify the previous placement of nodes to attain high quality phenomena monitoring. The required intelligence is fully distributed within the mobile sensor network so that the deployment algorithm is executed incrementally by different nodes. The presented algorithm adopts the Particle Swarm Optimization technique, which yields very promising results as reported in the paper (performance assessment). Our findings show that the proposed algorithm captures a certain phenomenon with very high accuracy while maintaining the network-wide energy expenditure at low levels. Random occurrences of similar phenomena put stress upon the algorithm which manages to react promptly and efficiently manage the available sensing resources in the broader setting.

Categories and Subject Descriptors: [**Computer systems organization**]: Embedded and cyber-physical systems – Sensor networks; [**Theory of computation**]: Distributed algorithms; [**Mathematics of computing**]: Bio-inspired optimization

Additional Key Words and Phrases: Mobile sensor networks, phenomenon localization, automatic node deployment, self-reorganization, computational intelligence, particle swarm optimization.

ACM Reference Format:
Christos Anagnostopoulos, Stathes Hadjiefthymiades, Kostas Kolomvatsos, 2015. Accurate, Dynamic, & Distributed Localization of Phenomena for Mobile Sensor Networks. *ACM Trans. Sensor Netw.* V, N, Article A (January YYYY), 60 pages.
DOI: http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

1.1. Motivation

Our motivation comes from the domain of (i) environmental monitoring, e.g., forest fire detection [Zervas et al. 2011], and (ii) dynamic remote monitoring systems, e.g., [Sayed-Mouchaweh and Lughofer 2012]. We assume the availability of motorized mobile nodes (e.g., robots) with mounted sensors (e.g., temperature, humidity, vision, light, air pressure, directional sound) that aim to accurately capture the current status of the contextual environmental parameters, e.g., fire front and gas leak, and trigger remedial actions. As such situations refer to dynamic phenomena occurrence, mobile

nodes need to constantly relocate (autonomously – not by external means) to better 'perceive' the new conditions in the changing environment [Sayed-Mouchaweh and Lughofer 2012]. In such event localization scenarios, nodes need to be dispatched close to the locations of the phenomena sources for performing the sensing task. Our primary objective is to 'bias' the spatial distribution of sensing elements so as to optimally capture the developments in the locations where phenomena take place. To better utilize the available resources, areas with no perceived phenomena will become coverage holes until the situation dynamically changes.

We focus on the class of **Mobile Sensor Network** (MSN) context-aware applications with the responsibility to detect and accurately monitor concurrent occurrences of phenomena (e.g., release of chemical plumes). The surrounding area of phenomena, referred to as **Region of Interest** (ROI), is the region in which the corresponding phenomena can be detected (measured) by sensors (robots). The phenomenon location is referred to as **Point of Interest** (POI). In our study, a ROI consists of a number of POIs. After a phenomenon detection, a group (swarm) of nodes is tasked to monitor the evolution of the phenomenon through the measurement of the corresponding physical parameter and, autonomously, approach the corresponding POI as much as possible. A POI may be a point representation of a certain phenomenon or lie in the boundary of a broader surface where the phenomenon takes place. For instance, in a forest fire, nodes surround the broader area of the POI and monitor the movement of the forest fire front while keeping themselves at a safe distance without being damaged.

In our work the relocation of a group of nodes towards the POIs is achieved by enabling dynamically selected nodes, namely *vanguards*, to make, locally, self-deployment decisions with respect to their neighbours. Such nodes give directives to their neighbours for optimal coverage of the regions surrounding the unknown POIs until the corresponding phenomena are fully covered (spatially).

1.2. Application Domains

Remote environmental monitoring and event localization in a ROI (e.g., fire, flood) are very important context-aware applications of wireless sensor networking. Spatially distributed mobile sensor nodes are used to monitor environmental contextual parameters (e.g., temperature, chemical compounds), locate the sources/events, and possibly transmit their data through the network. There has been a tremendous increase of interest in mobile robots and their context-aware applications. These robotic devices can be: Unmanned Ground Vehicles (UGV), Unmanned Aerial Vehicles (UAVs), Unmanned Underwater Vehicles (UUVs), or Unmanned Surface Vehicles (USVs).

Nowadays, remote monitoring, remote experimentation, and events localization achieved by mobile robots has opened new opportunities in context- and situation-aware applications with unknown and hazardous environments. Context- and situation-aware applications [Anagnostopoulos et al. 2008] utilize wireless mobile robots to detect several phenomena in unknown environments. Using mobile robots equipped with sensors is becoming increasingly popular, especially in environments where human intervention is limited, impossible, or dangerous. These context-aware robots can be used to perform POI localization and, subsequently, dangerous tasks in hazardous environments. Mobile robots can outperform human operators since, they are able to conduct measurements while exposed to hazardous conditions/substances, for example areas with high H_2S concentrations [Hernandez et al. 2012].

Indicative systems in the domain of remote monitoring that support machine intelligent applications and involve context-aware mobile robots are discussed below: The mobile robots in the system presented in [Hernandez et al. 2014] are equipped with gas sensors and remotely locate a gas/odour source. In this context the major tasks are gas detection (gas finding), odour discrimination and concentration estimation, gas source

tracking, and gas distribution mapping. Moreover, leak detection, emission monitoring and landfill monitoring in bio-gas facilities, more specifically landfill sites, are of critical importance and are gaining significant interest. This context-aware application refer to air pollution monitoring, where mobile robots are used for surveillance of landfill sites, and gas leak localization.

Moreover, situation-aware and monitoring systems like CReSIS[1] have developed ground and ice penetrating radars that can also provide highly detailed images of the subsurface in the polar region. Through these radars, the robots identify layers and areas of potential water at the interface between the ice sheet and the underlying bedrock. Together, these remote sensing techniques allow us to study and further understand how polar regions are changing in response to climate change [Gifford et al. 2010]. Finally, there are systems that combine robotics with remote sensing systems to autonomously image the subsurface for polar and planetary applications [Gifford et al. 2009].

1.3. Contribution & Organization

In this paper, we consider:

— a MSN of unknown number of nodes that are randomly spread on a ROI;
— each MSN node is energy constrained, is equipped with sensors, and monitors contextual environmental parameters (e.g., temperature, humidity);
— all nodes acquire only local information from their direct (1-hop) neighbours. Such information includes the current locations of neighbours and the magnitude of the sensed parameters;
— The nodes do not know the boundary of the ROI. Moreover, the exact location information for the POIs is previously unknown to the nodes;
— multiple phenomena randomly (both in time and space) appear in the sensing field.

Our approach assumes the performance of measurements of the physical quantity of the detected phenomenon and local evaluation. Based on such measurements, a group (swarm) of nodes locates certain phenomena by exchanging relocation directives among neighbours. If, at some time, another phenomenon (or phenomena) occurs, the group of nodes autonomously splits and different subsets of the initial group are self-deployed in order to cover the extra phenomena. Once groups of nodes target to the same phenomenon, they merge to one group.

The contribution of this work has as follows:

— we introduce a specific Particle Swarm Optimization (PSO) algorithm [Kulkani and Venayagamoorthy 2011] for *local*, *optimal* coverage of certain areas close to the vanguard nodes;
— we propose a reward-penalty fitness function, which is adopted by the *local* coverage process to yield the optimal location coordinates for the mobile nodes to further explore and sense a sub-region of the ROI;
— we propose a fully distributed, self-deployment, and relocation algorithm for optimal coverage of unknown POIs in a ROI with randomly occurring phenomena. The basic requirement for low energy expenditure is fully satisfied as the relocation of nodes is optimized;
— we propose two re-location policies for directing the mobile nodes where to sense and explore certain subregions of a ROI;

[1] https://www.cresis.ku.edu/

— we analyze the computational complexity of the proposed distributed algorithm and report on the expected cost in terms of computation cost (in-network processing), communication cost, and mobility cost of the nodes;
— we provide comprehensive sensitivity analysis and performance evaluation of the proposed algorithm by introducing five (5) realistic scenarios depending on the random appearance of multiple phenomena in a ROI;
— we provide an extensive experimental study on the consumed energy per node in terms of computation, sensing, communication and mobility energy;
— finally, we compare our coverage process with the Artificial Bee Colony (ABC) [Karaboga 2005, Karaboga and Akay 2009] computational intelligence algorithm in terms of the degree of coverage and execution time.

The structure of the paper is as follows. Section 2 gives a literature overview on the deployment schemes in MSNs and discusses the significance of our approach. In Section 3, we report on the preliminaries, the adopted sensing, connectivity, positioning models, and the phenomena representation model for dealing with the ROI coverage problem. In addition, Section 3 provides a background on the Particle Swarm Optimization (PSO) algorithm and on the adopted variant of the PSO algorithm, the Centroid PSO (CPSO). Section 4 formally defines the ROI coverage problem, while in Section 5 we propose the ROI coverage algorithm. In Section 6, we propose the distributed phenomena localization process, while in Section 7, we discuss the computational complexity and energy cost of the proposed process. Section 8 evaluates the proposed approach through certain performance metrics in different realistic scenarios and, also, reports on the energy consumption of our approach. In Section 9, we discuss candidate optimization algorithms from the Swarm Intelligence research area, namely the Ant Colony Optimization (ACO) and the Artificial Bees Colony (ABC). Moreover, we provide a comparative assessment of the adopted CPSO and the ABC algorithm in solving the ROI coverage problem. Finally, Section 10 concludes the paper and presents our future research agenda.

2. RELATED WORK

Many strategic MSN applications are used to improve centralized or decentralized coverage of ROIs [Gaojun and Shiyao 2010], [Gifford et al. 2010], through dynamic deployment algorithms [Dhillon and Chakrabarty 2003, Li et al. 2010, Wang et al. 2009]. Most of them focus on optimal deployment of the nodes in order to maximize the coverage performance. Self-deployment methods using mobile nodes [Wang et al. 2005, Wang et al. 2006, Heo and Varshney 2005] have been proposed to enhance ROI coverage and extend the system lifetime through uniformly distributed node topologies.

Significant research on sensor self-deployment algorithms for ROI coverage with no specific focus (i.e., with no localization of the event) exists in the literature. The Virtual Force (VF) algorithm [Zou and Chakrabarty 2004] is the most popular approach. Moreover, the algorithms in [Wang et al. 2010] and [Wang et al. 2007] adopt PSO and a combination of the VF algorithm with the co-evolutionary PSO for improving the effective coverage performance of the VF algorithm. In addition, the work in [Wan and Yi 2006] focuses on ROI coverage in which the locations of the nodes follow a 'Poisson point' process and sensors are uniformly distributed in the ROI. The authors in [Xiao et al. 2007] present a randomized scheduling algorithm where sensors are uniformly distributed. In [Megerian et al. 2005], the authors proposed an algorithm for ROI coverage calculation from the perspective of computational geometry. The algorithm in [Cortes et al. 2004] achieves ROI coverage through Voronoi polygons. The discussed algorithm is not distributed among nodes since links may be very long. In addition, the authors in [Bartolini et al. 2008] proposed a snap-and-spread self-deployment algorithm. In

this algorithm all nodes construct a hexagonal tiling by 'pushing' and 'pulling' nodes to hexagonal centres. The discussed algorithm is not distributed. The work in [Tu et al. 2012] discusses a self-deployment model which relocates nodes to surround a unique target of interest uniformly (static or moving event that progresses). The discussed model assumes that every node obtains the direction of the target and knows its dynamics (position, velocity, and acceleration).

Our approach differs significantly from previous work found in the literature. We deal with (i) a set of nodes of *unknown* number, which may be fixed or time-varying, (ii) an *unknown* number of POIs, and (iii) how such nodes are dynamically spread to localize the corresponding phenomena, which appear/disappear dynamically over space and time. The proposed distributed algorithm is not a pure ROI coverage algorithm but, *also*, deals with the localization of the corresponding POIs due to concurrent phenomena. Our approach introduces a fully distributed ROI coverage technique. Our scheme is based on local computation of the self-deployment decisions instead of the global, centralized calculation introduced by previous models. Through local computations, the ROI coverage and POI localization are incrementally achieved by various nodes, thus, reflecting the distributed intelligence nature of the scheme. Our approach does not assume any initial topology of nodes and results to the distribution of nodes around each POI.

The works closest to our approach are: the work in [Garetto et al. 2007] and the F-coverage formation problem discussed in [Li et al. 2010]. The model in [Li et al. 2010] covers a specific focus area around a *known* POI. This is achieved by positioning nodes at certain polygon layers. Such model is based on geometric rules for moving nodes around the unique globally known POI. As previously stated, our approach deals with *unknown* POIs (unknown in number and location), which correspond to certain real life phenomena. Through this realistic setting, nodes freely move to optimally cover the corresponding ROIs and locate phenomena with the needed accuracy.

The work in [Garetto et al. 2007] studies the sensor deployment and relocation issue through the virtual forces paradigm proposed in [Zou and Chakrabarty 2003]. Specifically, the work in [Garetto et al. 2007] makes some important assumptions on the characteristics of the MSN. Nodes should be equipped with technologies that not only determine their location but also the direction of arrival of messages transmitted by their peers in the MSN. Additionally, the problem of determining the orientation to the location of an incident (POI) is treated through a centralized manner and imposes extra node movement (that could exhaust the nodal energy budget). Lastly, the total complexity of the solution (over all the network nodes) is quite high as the whole phenomena monitoring 'group' (and, possibly, more nodes) should perform similar calculations. In our model there is no need to determine the orientation to the location of the phenomenon (thus, reducing the technical complexity of the node). Moreover, the adoption of the vanguard role (with potential substitutions throughout the phenomena coverage phase) reduces the overall complexity of the solution (vanguard performs calculations and instructs its peers with $O(1)$ time and communication complexity as will be discussed later) and the associated signalling load. Finally, our method is fully distributed, i.e., there is no central entity for determining and disseminating relocation directives to the nodes to optimally cover and localize POIs. Such relocation directives are locally determined by vanguard nodes in a rolling assignment fashion, and through a novel domino-like process (explained later), the nodes are engaged in certain swarms moving towards the POIs.

2.1. Significance of Our Approach

In this section, we mention the significance of our approach for the distributed localization phenomena problem, which further depicts the differences to the related works

found in the literature. Firstly, our approach is fully distributed and exploits strictly local knowledge and decision making in light of relocation directives. Specifically, a significant ingredient of our approach is the local election process. Such a process provides the local control for relocation directives by electing the vanguard node among neighboring nodes. The election process terminates in $O(1)$ iterations and requires $O(1)$ message exchanges (see Section 6.3) taking into consideration the residual energy budget of a node and the magnitude value of the phenomenon experienced by that node. Secondly, only one node in a neighborhood (i.e., the vanguard) executes the optimal coverage process (Section 5) based upon trivial knowledge: location coordinates of its neighboring nodes. Then, the vanguard simply broadcasts the relocation directives to its neighbors and, thus, a swarm of robots is (self)formulated with only $O(1)$ communication overhead (see Section 6.3.2). Thirdly, our 'domino-like' approach (Section 6.2) guarantees that the vanguard leadership is distributed among nodes and, in parallel, nodes as a controlled swarm, move towards the POI. Fourthly, the proposed two relocation policies (Section 6.2.1) provide the flexibility to a neighborhood to dynamically split or merge with other neighborhoods in light of moving towards multiple phenomena in the ROI. In addition, in our approach there is no assumption on the number of the phenomena, and more importantly, there is no knowledge on the locations of the phenomena. Nodes move around the ROI and, with only incremental local decisions, are able to self-organize when some of them firstly detect a phenomenon (also referred to as first responders in Section 6.1). It is also worth noting that there is no central entity that disseminates the relocation directives to the entire network, as previous approaches do. In our approach, the relocation directives are produced locally on the vanguards. In addition, the local coverage optimization algorithm is relatively simple and is based on the principles of the computational intelligence and the meta-heuristic PSO algorithm. The input to the optimization process, in Section 5, is only the location coordinates of the neighboring nodes, while during the coverage optimization process no communication is required and no sensed contextual information is disseminated within a neighborhood. Such meta-heuristic optimization algorithm is adopted because, apart from its simplicity and capability of providing an optimal solution in high dimensional spaces (in our case, the dimension of the search space is twice the number of the neighboring nodes), it also supports any arbitrary optimization function, also known as fitness function (Section 5.1). This gives us the flexibility to define our own objective/fitness function to get the optimal placement of the neighboring nodes in a sub-region of the ROI. Moreover, a detailed discussion and comparative assessment with other meta-heuristic optimization algorithms (ACO and ABC) is provided in Section 9. Nonetheless, we show that the adopted and modified PSO algorithm is the most efficient in terms of the degree of coverage and computational complexity with respect to our fitness function for optimal coverage.

3. PRELIMINARIES

In this section, we report on certain theoretical models for the sensor coverage probability, sensor connectivity, sensor positioning mechanism, the phenomenon representation and discuss their adoption to our method. Moreover, we provide preliminaries on the PSO adopted for materializing the ROI coverage problem.

3.1. Sensor Coverage Model

A sensor converts physical stimuli into electrical or other recordable signals. These signals are further processed to output digital sensing data which carry comprehensible information. Many measurement mechanisms have been used to quantify how well sensors work in terms of sensing. The sensor characteristics, such as transfer function, sensitivity, dynamic range, and manufacturer accuracy, can be used to measure how

well a sensor reacts to physical stimuli [Wang 2010]. In our case, the adoption of a *sensor coverage model* is required as a mechanism to measure sensors' sensing capability and quality for the Remote Monitoring Context-Aware Applications (RMCAA) since we are interested in the optimal placement of the nodes in an ROI for better phenomena capturing.

Sensor coverage models reflect the sensing capability and quality of sensors [Wang 2010]. In most cases, a sensor coverage model is mathematically formulated as a function of the distance between a point and the location of the sensor node. The output of this function is a non-negative scalar indicating a coverage measure of this point, e.g., a probability measure. This abstraction in light of the distance between sensor location and the phenomenon/event/source location makes it ideal in our POI phenomena localization problem.

The degree of coverage at a specific point in a ROI is related to the number of sensors whose sensing range cover that point. It has been observed and postulated that different RMCAA would require different degrees of coverage in the ROI. For instance, a military surveillance RMCAA would need a high degree of coverage, because it would require a region to be monitored by multiple nodes simultaneously, such that even if some nodes cease to function, the security of the region will not be compromised. Whereas, some of the environmental RMCAAs, such as animal habitat monitoring, temperature monitoring inside a building, and forest fire monitoring, might require a relatively low degree of coverage.

Before discussing certain sensor coverage models, we firstly report on the disc model [Meguerdichian et al. 2001, Xing et al. 2005] for representing the sensing and connectivity capability of a mobile node. A ROI is defined as a bounded convex region $F \subset \mathbb{R}^2$ in which nodes are deployed to detect any phenomenon that occurs within F. An epicentre ROI is defined as a disc $F(q,r) \subset \mathbb{R}^2$ with center $q \in \mathbb{R}^2$ and radius $r > 0$. Consider a set of mobile nodes randomly deployed in F. Each node is indexed through a positive integer $i \in \mathcal{N}$ and the number of nodes n may be fixed, $n = |\mathcal{N}|$, or time-varying, $n(t)$, if nodes are allowed not to operate or new nodes are introduced. Each node has current location $\ell_i = (x_i, y_i) \in F$. Each node is equipped with a set of sensors. It measures physical quantities that are connected to physical phenomena.

As mentioned above, a sensor coverage model is mathematically formulated by a coverage function. We have distinguished two types of coverage functions: (i) the binary coverage function, where the coverage measure is either 0 or 1 for a point and (ii) general coverage function, where the coverage measure can take various nonnegative values in $[0, 1]$. The sensing capability (coverage performance) of a node corresponds to the region swept by a disc around the node. For simplicity reasons, we assume that all nodes have the same *sensing range* δ; however this does not affect the proposed scheme. For any point $q = (x, y) \in F$, we notate the Euclidean distance from node i as $\| \ell_i - q \| = \sqrt{(x - x_i)^2 + (y - y_i)^2}$.

Let $\theta_q(\ell_i)$ be a coverage measure quantifying the capability of the node i to cover a point $q \in F$. The binary coverage function in [Chakrabarty et al. 2001, Chakrabarty et al. 2002] expresses the coverage measure as the probability of coverage $\theta_q(\ell_i) \in \{0, 1\}$ of a point $q \in F$ as follows:

$$\theta_q(\ell_i) = \begin{cases} 1, \text{ if } \| \ell_i - q \| < \delta \\ 0, \text{ otherwise.} \end{cases} \quad (1)$$

In reality, sensor detections are imprecise, thus, the authors in [Dhillon and Chakrabarty 2003, Zou and Chakrabarty 2004, Zou and Chakrabarty 2005] define a generic coverage function, which is introduced in [Elfes 1990], through which the

probability of coverage $\theta_q(\ell_i) \in [0,1]$ also 'integrates' such imprecision, i.e.,

$$\theta_q(\ell_i) = \begin{cases} 0, \text{ if } \delta + \delta_e \leq \| \ell_i - q \| \\ \exp(-\alpha_1 \left(\frac{\lambda_1^{\beta_1}}{\lambda_2^{\beta_2}} \right) + \alpha_2), \text{ if } \delta - \delta_e < \| \ell_i - q \| < \delta + \delta_e \\ 1, \text{ if } \delta - \delta_e \geq \| \ell_i - q \| . \end{cases} \quad (2)$$

The δ_e parameter is a measure of imprecision in node detection capability, $\lambda_1 = \| \ell_i - q \| - \delta + \delta_e$, $\lambda_2 = \delta + \delta_e - \| \ell_i - q \|$, and $\alpha_1, \beta_1, \beta_2$ are parameters that measure detection probability when point q is at distance greater than δ_e but within distance from the node. The α_2 parameter is a disturbing effect. The values of the parameters $\alpha_1, \beta_1, \beta_2$ can be interviewed as the characteristics of various types of physical sensors [Dhillon and Chakrabarty 2003]. The reason for adopting the sensor coverage model in (2) in our case is that this model reflects the behaviour of range sensing devices such as infrared and ultrasound sensors; the sensor placement and localization algorithms are independent of the sensor sensing models. Moreover, as discussed later, this general coverage function provides the flexibility to quantify how well a node covers a specific area. This information is highly exploited by our ROI coverage algorithm (especially in evaluating the PSO fitness function as discussed in Section 5.1) in the coverage optimization problem. It is worth noting that we can adopt alternative sensing models that are based on radio signal propagation models in which signal strength decays as a power of distance. In addition, since we deal with the phenomena localization problem, this model incorporates the (unknown) location of the POI, which we are attempting to identify (estimate). Finally, the determination of the relocation directives, in our method, is mainly driven by the probability of coverage (coverage transfer function) $\theta_q(\ell)$. Hence, such model is highly desirable to be adopted in our method. Given a sensor coverage model with $\theta_q(\ell) \in [0,1]$, we define the sensing region of a node as follows:

Definition 3.1. The sensing region $F(\ell_i, \delta)$ of a node i is the set of those points $q \in F$ such that the sensing performance of node i is better than that of any other node, i.e.,

$$F(\ell_i, \delta) = \{q \in F : \| q - \ell_i \| \leq \delta \text{ and } \theta_q(\ell_i) \geq \theta_q(\ell_j), \forall j \neq i, j \in \mathcal{N}\}. \quad (3)$$

Furthermore, the total coverage probability of a point q, also referred to as 1-coverage probability, from a set of nodes $\mathcal{N}$ is $1 - \prod_{i \in \mathcal{N}}(1 - \theta_q(\ell_i))$, i.e., the probability that point q is covered by at least one sensor node.

3.2. Sensor Connectivity Model

Area coverage and connectivity in a MSN are interdependent problems [Xu et al. 2006]. Accordingly, our goal is an optimal sensor deployment in certain POIs where nodes form a locally connected network (through islands formation as discussed later in subsequent paragraphs) while optimizing the coverage of multiple POIs at the same time. By optimizing coverage, the deployment strategy would guarantee that the largest possible area in the sensing field of multiple POIs is covered by sensors, as required by the underlying RMCAA. By considering that an island of nodes is connected, we ensure that certain information is transmitted to the neighbouring nodes (certain relocation directives as discussed later) in light of the phenomena localization process.

In our model, each node makes self-deployment decisions independently using 1-hop neighbourhood location information. Two nodes can communicate with each other only if the distance between them is less than a *communication range* ϵ.

Definition 3.2. At some time instance, the neighbours of node i belong to set $\mathcal{N}_i$:

$$\mathcal{N}_i = \{j \in \mathcal{N} | \ \| \ell_i - \ell_j \| < 2\epsilon, j = 1, \ldots, n, j \neq i\} \quad (4)$$

This connectivity model is ideal for our case since we require 1-hop communication between a vanguard and its neighboring nodes for disseminating the optimal relocation directives. Also, the cluster-based election process of a vanguard (discussed later) relies on the communication range ϵ and through the definition of the nodes neighbourhood in (4), we achieve computational and communication efficiency. Finally, the proposed fitness function for the considered optimization problem (discussed later) depends on the ratio ϵ/δ, which refers to the area ratio of the communication and sensing disc models, respectively.

There is an interesting relation between ϵ and δ [Xu et al. 2006]. For instance, the 'coverage-first placement' method in [Wang et al. 2008] attempts to reduce the number of nodes by minimizing the overlapping coverage. Such placement is efficient when $\epsilon \geq \sqrt{3}\delta$ since connectivity is guaranteed. More advanced placement requirements can also be applied [Wang et al. 2008]. In our case, there is no constraint on the continuous connectivity among all nodes of the network, since there is no requirement of information dissemination within the entire MSN. Specifically, in our method there is no need to disseminate the measurements throughout the MSN to achieve optimal coverage of the ROI. Instead, only local message exchanges (within 1-hop communication), which involve relocation directives, are required, thus, making our communication method efficient for the considered problem. We require a specific number of nodes (forming a swarm of nodes) to cover and locate at least one POI and, then, sharing certain relocation directives for improving the localization of the observed phenomena. However, we could define a minimum number of nodes to be always connected in such a neighbourhood / formation [Bai et al. 2008]; see also Section 6.2.2.

3.3. Sensor Positioning Model

Self-localization capability is a desirable feature of MSNs. In our case, location information (relative or absolute) or orientation information is required by the optimal relocation process (see Section 6.2.1) to dynamically position a swarm of nodes towards the ROI coverage and POIs localization. Several approaches in location discovery in WSNs assumes the availability of GPS receivers at some nodes or beacon nodes with known position [Bulusu et al. 2004, Wu et al. 2005, Yu et al. 2006]. Their locations are then used to determine the positions of other sensor nodes, which do not have GPS receivers. Moreover, relative localization methods are applied, where nodes, regardless of their absolute coordinate (location) knowledge, estimate the range between themselves and their neighboring devices. For instance, there are localization techniques that exploit the received signal strength indicator of the RF communication, and the time difference in arrival of RF and acoustic (ultra-sound) signals. Once a node knows its distance to three non-collinear landmark nodes, it can estimate its location by trilateration[2]. The interested reader could refer to the survey in [Savvides et al. 2001] which provides detailed discussion on location discovery algorithms in MSN and the GPS-free localization model in [Wang & Xu 2010]. In our case, we assume that nodes (i) either know their position directly through GPS or (ii) they could adopt other localization techniques [Alcan et al. 2010, Esnaashari and Meybodi 2011] or (iii) a hybrid scheme, as, in our model, location information from neighbouring nodes (either absolute or relative w.r.t. the position of the vanguard) is exploited for the relocation directives.

[2]IPAC: Integrated Platform for Autonomic Computing (NFSO-ICT-224395), Deliverable document D5.1b, 'IPAC Trial Report, Industrial Environment', April, 2011 [http://ipac.di.uoa.gr/]

3.4. Phenomenon Model

A phenomenon ϕ is a detectable event through the measurement of a physical quantity, e.g., temperature, humidity, pressure, heat flux, chemical concentrations. Such quantity is represented as a spatio-temporal field along with a location-dependent *magnitude* value. The disc model is well applied once the spatial phenomena involve local changes in gradients of physical quantities.

Let Φ be a set of phenomena $\phi \in \Phi$ and a phenomenon ϕ initiated at a specific location $\ell_\phi \in F$. We assume that phenomena occur randomly in space and time. Consider also discretized time slots $t \in \mathbb{T}$. We notate $\mu_{\phi,i}(t)$ as the magnitude of the phenomenon ϕ, which is measured (sensed) by node i located at ℓ_i at time t. We consider phenomena whose magnitude values measured at nodes decrease with the distance from the nodes [Chandy et al. 2010], [Manolakos et al. 2008], i.e., the $\mu_{\phi,i}(t)$ value decreases with the distance $\parallel \ell_i - \ell_\phi \parallel$. Let $\mu_{\phi,i}(t) \in [0, \mu_{\max}]$, where a value of zero indicates that the phenomenon ϕ is not detected by node i, while $\mu_{\max}$ indicates that the node i locates very close to ϕ. Furthermore, in general, there are many parameters that may affect the measurement result. The way they impact the measurement result can, in many cases, be mathematically modeled as in [Adams 2002]. It is worth noting that, in this paper, we do not deal with inaccuracies and the uncertainty of measurement due to the sensor circuitry. We assume that the sensor i is supported by certain mechanisms to handle such uncertainty [Helm et al. 2010]. We base our framework on the the derived magnitude $\mu_{\phi,i}$ corresponding to a phenomenon ϕ.

In the literature, e.g., [Chandy et al. 2010], the magnitude value $\mu_{\phi,i}$ decreases with the distance $\|\ell_i - \ell_\phi\|$. For instance, for a nuclear radiation sensor located at ℓ_i and for a source (phenomenon) ϕ located at ℓ_ϕ, which generates p_1 particles per unit time t, the rate of photons from the source detected by that sensor is $\mu_{\phi,i} = \frac{p_1 p_2 e^{-p_3 \|\ell_i - \ell_\phi\|}}{\|\ell_i - \ell_\phi\|^2}$, where p_3 is the photon absorption rate per meter, in air and p_2 is a proportionality constant which depends on factors such as the size of the sensor's detection crystal [Chandy et al. 2010]. In this paper, we adopt the magnitude value proposed in [Manolakos et al. 2008] (radiation intensity)

$$\mu_{\phi,i}(t) = m_1(1 - \exp(-\frac{m_2}{\parallel \ell_i - \ell_\phi \parallel})). \tag{5}$$

The parameters m_1 and m_2 refer to certain constants [Manolakos et al. 2008] that affect the sensed magnitude value. The proposed phenomenon representation works under a wide class of magnitude functions, which are distance-dependent and, especially, distance-decreasing. It is worth noting that we could adopt other distance-dependent magnitude models since they do not limit the applicability of our algorithms and the distributed mechanism for phenomena localization. We adopt the model in [Manolakos et al. 2008] since it is applied on the RMCAA domain, which motivated us for this work.

3.5. Particle Swarm Optimization

The Particle Swarm Optimization (PSO) algorithm [Kennedy and Eberhart 1995, Eberhart and Kennedy 1995] mimics swarm behaviour in birds flocking and fish schooling to guide particles to search for globally optimal solutions. The PSO algorithm searches a space by adjusting the trajectories of individual vectors referred to as *particles*. The particles are moved stochastically toward the positions of their own previous best performance and the best previous performance of their neighbours. A swarm S consists of a set of m particles. Each particle represents a potential solution in a D-dimensional space. A particle i is associated with a velocity vector $\mathbf{v}_i = [v_i^1, v_i^2, \ldots, v_i^D]$ and a position vector $\mathbf{p}_i = [p_i^1, p_i^2, \ldots, p_i^D]$. A population of particles is initialized with random positions $\mathbf{p}_i$ and velocities $\mathbf{v}_i$. A *fitness* function $J : \mathbb{R}^D \to \mathbb{R}$ is evaluated by

using a particle's position $\mathbf{p}_i$ as input value. Positions and velocities are adjusted and $J(\mathbf{p})$ is evaluated with the new coordinates at each time step.

When a particle i discovers a solution that is better than any other solution previously determined (w.r.t. $J(\mathbf{p})$), it locally stores the coordinates in a vector $\mathbf{p}_{\ell i}$ (the local best point found by the particle i so far). The difference between $\mathbf{p}_{\ell i}$ and the current $\mathbf{p}_i$ is stochastically added to the velocity $\mathbf{v}_i$ causing the trajectory to oscillate around that point. Each particle is associated with a topological neighbourhood including itself and some other particles in S (possibly all), notated by V_i. The stochastically weighted difference between the neighbourhood's best position $\mathbf{p_g}$ (i.e., $\mathbf{p_g} = \arg\max_{j \in V_i}\{J(\mathbf{p}_j)\}$) and the particle's current $\mathbf{p}_i$ is also added to its velocity, adjusting it for the next step. Such adjustments to the movement of the particle cause it to search around the two best positions, i.e., $\mathbf{p}_{\ell i}$ and $\mathbf{p_g}$. The particle-specific Centroid PSO (CPSO) algorithm [Liang et al. 2010] improves the global optimum efficiency and the accuracy compared to PSO. The centroid of a neighbourhood V_i at time t is $\mathbf{p_c}(t) = \frac{1}{|V_i|}\sum_{i \leq |V_i|} \mathbf{p}_i(t)$. In this case, the trajectory of each particle is not only associated with the local and the neighbourhood best position, but also with the local best positions of other particles too.

During the evolutionary process of the CPSO, the velocity v_i^d and position p_i^d of particle i on dimension $d = 1, \ldots, D$ are updated at time t as follows:

$$\begin{aligned} v_i^d(t) &= w(t)v_i^d(t-1) + c_1 r_1^d(t)(p_{\ell_i}^d(t) - p_i^d(t)) \\ &+ c_2 r_2^d(t)(p_g^d(t) - p_i^d(t)) \\ &+ c_3 r_3^d(t)(p_c^d(t) - p_i^d(t)) \end{aligned} \tag{6}$$

and

$$p_i^d(t+1) = p_i^d(t) + v_i^d(t) \tag{7}$$

where $w(t)$ is the inertia weight [Shi and Eberhart 1998], c_1, c_2 and c_3 are the acceleration coefficients [Eberhart and Kennedy 1995], and $r_1^d(t), r_2^d, r_3^d \sim U(0,1)$, where $U(0,1)$ denotes the uniform distribution in [0,1]. A user-specific parameter $v_{max}^d \in \mathbb{R}^+$ is applied to limit the maximum velocity of each particle on the d-th dimension; if $|v_i^d| > v_{max}^d$ then $v_i^d = sign(v_i^d)v_{max}^d$. We adopt the maximum velocity v_{max}^d as the 20% of the search range proposed in [Eberhart and Shi 2001]. The inertia weight w decreases linearly [Shi and Eberhart 1998] with the iterations (steps):

$$w(t) = w_{max} - (w_{max} - w_{min})\frac{t}{T}, \tag{8}$$

where $t \in \{0, 1, \ldots, T\}$ is the iteration index and T is a predefined maximum number of iterations. For the parameters $\psi_k(t) = c_k r_k(t), k = 1, \ldots, 3$, it holds true that

$$\sum_k \psi_k(t) \leq 1 + w(t) + \sqrt{2 + 2w(t)} \tag{9}$$

in order for the CPSO to converge [Linag et al. 2010]. The adopted CPSO algorithm is shown in Algorithm 1. The algorithm returns the global best position $\mathbf{p_g}$, which is the solution for maximizing the $J(\mathbf{p})$ function. Note, the time complexity for the CPSO depends on the number of iterations T, the number of particles m and the time complexity $O(J)$ of the fitness function $J(\mathbf{p})$, i.e., $O(m \cdot T \cdot O(J))$. In Section 5.1.2, we elaborate on the time complexity of $J(\mathbf{p})$, thus, providing details of the adopted CPSO time complexity.

ALGORITHM 1: The CPSO Algorithm.

Input: Swarm $S = \{(\mathbf{p}_i, \mathbf{v}_i)\}_{i \leq m}$, maximum number of iterations T.
Output: Optimal solution $\mathbf{p_g}$.
$t \leftarrow 0$ /*iteration index*/ ;
while $t \leq T - 1$ **do**
  for *each particle* $i \in S$ **do**
    if $J(\mathbf{p}_i) > J(\mathbf{p}_{\ell_i})$ **then**
      /*update the local best solution for particle i*/;
      $\mathbf{p}_{\ell_i} \leftarrow \mathbf{p}_i$;
    end
    /*update the neighborhood best solution for particle i*/ ;
    $\mathbf{p_g} = \arg\max_{j \in V_i} \{J(\mathbf{p}_j)\}$;
    /*update the centroid of the swarm*/ ;
    $\mathbf{p_c} = \frac{1}{|V_i|} \sum_{i \leq |V_i|} \mathbf{p}_i$;
    /*update the current position and velocity of each particle */ ;
    Update $\mathbf{v}_i$ using Eq(6);
    Update $\mathbf{p}_i$ using Eq(7);
  end
  $t \leftarrow t + 1$;
end

4. PROBLEM FORMULATION

In this section, we formulate the problem of a ROI coverage. As discussed in Section 3.1, different MSNs applications require 'multilevel' coverage where each point in a ROI is covered by more than one node. In the k-coverage problem, each point of a ROI should be covered by at least k nodes, $k \geq 1$. For instance, if the sensing range of two nodes overlap, i.e., a region is within the sensing range of two nodes, the overlapped region is called 2-covered. The k-coverage helps to design a fault-tolerant and robust MSN. In addition, redundant k nodes can ensure more reliable sensor data received from the environment.

Certain coverage problems have been addressed in different research fields. The Art Gallery problem [Marengoni et al. 1996] and the Disc Covering problem [Williams 1979] of computational geometry are related to the coverage problem. The Art Gallery problem (we elaborate on this in Section 6.2.2) refers to the minimum number of observers required to monitor a polygon area assuming that an observer can watch all the points within its line of sight. The Disc Covering problem asks for the smallest radius of n identical discs which can be arranged to cover a unit disc. However, solutions of these problems are not directly applicable to MSNs applications due to the nature of sensor nodes. For example, sensor nodes have limited sensing range and battery life whereas the observers of the Art Gallery problem have infinite visibility unless an obstacle appears. Unlike the observers, all the sensor nodes need to communicate among each other within its limited communication range in MSNs.

Two categories of k-coverage problems have been identified: (i) The k-coverage verification: In this category, the k-coverage problem is formulated as a decision problem where a ROI needs to be verified whether it is k-covered or not. A baseline solution to this decision problem [Huang & Tseng 2005] comes with $O(n^3)$ complexity where n is the number of neighboring nodes within the sensing radius of a node. (ii) The nodes subset selection k-coverage problem: For a desired coverage degree, $k \geq 1$, select a minimal subset of already deployed nodes in a ROI so that every point of the ROI is within the sensing range of at least k different sensors. It has been proved that this

k-coverage problem is NP-hard by reduction to the minimum dominating set problem [Yang 2006].

In our case, we do not pursue the (global) k-coverage problem for phenomena localization in a ROI with $k > 1$. Instead, for a given node i, we locally gather around its sensing region a certain number of nodes to collaboratively improve the coverage outside the sensing region of node i, which is equivalent to a 1-coverage local problem. This enables a biased dynamic relocation of neighboring nodes towards the POI of one or more phenomena. This dynamic nature of our problem cannot be treated as an instance of a *static* k-coverage problem, since nodes gradually relocate to new areas directed to the phenomena POIs. However, a discussion on the minimum number of neighboring nodes and on a local coverage degree $k \geq 1$ is provided in Section 6.2.2 based on the Art Gallery problem.

Consider a scalar density function $h : F \to \mathbb{R}$, which assigns to $q \in F$ the importance measure of covering it. $h(q)$ satisfies $h(q) > 0$ for all $q \in F$ and $\int_F h(q)\, dq < \infty$. We have that $h(q) = 0$ for $q \notin F$ because the nodes ignore all points outside F. We define the coverage utility function $\mathcal{H}_F$ over F with n sensing regions $F(\ell_i, \delta) \subset F, i = 1, \ldots, n$, corresponding to sensor nodes as follows:

$$\mathcal{H}_F(\ell_1, \ldots, \ell_n) = \sum_{i=1}^{n} \int_{F(\ell_i,\delta)} h(q)\theta_q(\ell_i)dq \tag{10}$$

$\mathcal{H}_F$ is the cumulative sensing performance of all nodes over their sensing region including $h(\cdot)$.

Problem 1. The ROI coverage problem refers to finding the set of locations $(\ell_1^*, \ldots, \ell_n^*)$ of all n nodes that maximize $\mathcal{H}_F$:

$$\begin{aligned}(\ell_1^*, \ldots, \ell_n^*) = \arg \max_{(\ell_1,\ldots,\ell_n)} \mathcal{H}_F(\ell_1, \ldots, \ell_n)\\ \text{s.t. } \ell_i \in F, i = 1, \ldots, n\end{aligned} \tag{11}$$

We consider $h(q)$ as the indicator function $h(q) = \mathbf{1}_F(q)$ since all points $q \in F$ are of equal importance to be covered due to random phenomena occurrences inside F. The same problem definition as in Problem 1 is defined for $F(q, r)$ coverage; F is replaced with $F(q, r)$ in Eq(11). In this case, we assume that $r > \delta$. Note here that, the Problem 1 does not involve any restriction on the minimum number of nodes to cover at least a specific point, as in a k-coverage problem. Instead, one should maximize the probability of coverage of a given ROI only with the *available* number of nodes. This reflects the nature of our problem where, it is possible that the number of the neighboring nodes n of a node i will not be adequate for securing e.g., a degree of coverage $k > 1$ of the broader area of node i; see also Section 6.2.2.

Consider that a node i detects a phenomenon within its sensing region $F(\ell_i, \delta)$. Node i improves its local environmental awareness by gathering the neighbours $\mathcal{N}_i$ to a broader area of its sensing region. The relocation of the neighbours results to a finer detection in the area, thus, maximizing the probability of more accurate phenomenon localization.

Let $F(\ell_i, \epsilon)$ be the communication region of node i which communicates with $\mathcal{N}_i$ neighbours. The $n_i = |\mathcal{N}_i|$ neighbours are relocated to optimally cover the $F(\ell_i, \zeta)$ with $F(\ell_i, \delta) \subset F(\ell_i, \zeta) \subset F(\ell_i, \epsilon)$, in order to further sense the area around node i with $\delta < \zeta \leq \epsilon$. If the optimal coverage of $F(\ell_i, \zeta)$ requires a lower number of nodes than $|\mathcal{N}_i|$, then $n_i < |\mathcal{N}_i|$. On the other hand, if the number of neighbours is not adequate to

cover $F(\ell_i, \zeta)$ then the proposed approach could adjust the ζ radius to meet the minimum required number of nodes for the full coverage of $F(\ell_i, \zeta)$. In this work we assume that $n_i = |\mathcal{N}_i|$ is the adequate number of nodes for optimal coverage of $F(\ell_i, \zeta)$. A detailed analysis on the minimum number of nodes to cover a specific area is provided in Section 6.2.2. If $\zeta > \epsilon$ then node i 'spreads' its neighbours outside its communication range $F(\ell_i, \epsilon)$ for further area inspection. This may lead to the isolation of node i and thus the inability to be notified on a phenomenon occurrence. Hence, in that case, the swarm of nodes cannot communicate and, thus, efficiently identify and 'monitor' the phenomenon evolution. On the other hand, a value of $\zeta < \delta$ is not reasonable since node i detects phenomena within its region.

5. THE ROI COVERAGE ALGORITHM

The ROI coverage algorithm refers to a local algorithm which is executed on a selected node i located in ℓ_i adopting the CPSO algorithm. The major part of this algorithm is the fitness function $J(\mathbf{p})$. In this section, we also provide an analysis on the basic parameters of the ROI converge algorithm which influence the optimal solution of the Problem 1.

The CPSO algorithm is executed on a node i in order to relocate its neighbours to the $F(\ell_i, \zeta)$ area around it. The $F(\ell_i, \zeta)$ area is treated as a two-dimensional grid. The granularity of the grid, i.e., the distance between grid points, can be adjusted to balance the computation time of evaluating $J(\cdot)$ with the effectiveness of the coverage measure (a more detailed discussion follows in Section 5.1.2). We abstract the location coordinates $\ell_j = (x_j, y_j) \in F(\ell_i, \epsilon)$ of all neighbours $j \in \mathcal{N}_i$ as a particle in the swarm S in a $D = 2 \cdot n_i$ dimensional space with position vector

$$\mathbf{p} = [x_1, y_1, x_2, y_2, \ldots, x_{n_i}, y_{n_i}]^\top. \tag{12}$$

Let m be the number of particles in swarm S, i.e., the population size. The node i executes the CPSO algorithm in order to cover the $F(\ell_i, \zeta)$ (thus, maximizing $\mathcal{H}_{F(\ell_i,\zeta)}$ in Eq(10)). The optimal solution forms a cycloid-like topology of the locations of the neighbours within $F(\ell_i, \zeta)$ around ℓ_i as shown in Figure 1. After the CPSO execution by the node i, its neighbours $\mathcal{N}_i$ are relocated with respect to the global best solution $\mathbf{p_g}$:

$$\begin{aligned} \mathbf{p_g} &= [x_1^*, y_1^*, x_2^*, y_2^*, \ldots, x_{n_i}^*, y_{n_i}^*]^\top \\ &= \arg \max_{\mathbf{p} \in F(\ell_i,\zeta)^D} \mathcal{H}_{F(\ell_i,\zeta)}(\mathbf{p}) \end{aligned} \tag{13}$$

Furthermore, we are interested in minimizing the relocation (moving) distance, thus, reducing the mobility energy consumption, of a neighbour $j \in \mathcal{N}_i$, which moves from its initial location $\ell_j \in F(\ell_i, \epsilon)$ (before the CPSO execution) to the optimal location $\ell_j^* = (x_j^*, y_j^*) \in F(\ell_i, \zeta)$. The neighbour j relocates to ℓ_j^* such that

$$\ell_j^* = \arg \min_{(x^*, y^*) \in \mathbf{p_g}} \| (x^*, y^*) - (x_j, y_j) \| . \tag{14}$$

In Section 7 we report on the mobility energy consumption and its impact on the overall performance of the proposed process.

5.1. The ROI Fitness Function

The basic part of the ROI coverage algorithm is the ROI fitness function $J : \mathbb{R}^D \to \mathbb{R}$ of the CPSO algorithm. The $F(\ell, \zeta)$ coverage maximization is performed by minimizing the areas which are not covered by the sensing region $F(\ell_i, \delta)$ of any node i within $F(\ell, \zeta)$ as shown in Figure 1. We propose a reward-penalty based function, $J(\mathbf{p})$, which evaluates the candidate solution $\mathbf{p} \in \mathbb{R}^D$ in terms of the $F(\ell, \zeta)$ coverage. The input of

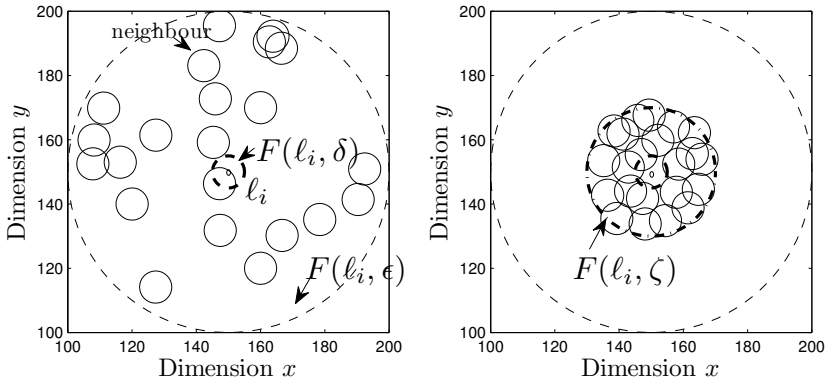

Fig. 1. The $F(\ell_i, \zeta)$ coverage with $n = 22$ neighbours and $m = 20$ particles ($T = 20$ iterations); (left) before the CPSO execution, (right) after the CPSO execution.

the $J(\mathbf{p})$ function is a possible (candidate) solution $\mathbf{p}$ for a specific coverage of $F(\ell, \zeta)$ initiated by a node located at ℓ with connectivity range $\epsilon > \zeta$.

5.1.1. Fitness Function Rationale. The rationale of the $J(\mathbf{p})$ function is to evaluate whether the placement of the nodes within the $F(\ell, \zeta)$ is such that every point q within this area is covered by (at least) one node, i.e., referring to the instance of the 1-coverage problem. To this end, for each point $q \in F(\ell, \zeta)$, we introduce a certain penalty for the candidate solution $\mathbf{p}$ if this point q is not covered by any node. On the other hand, we require that all nodes are uniformly spread in the $F(\ell, \zeta)$ area and not get concentrated in certain sub-areas of the $F(\ell, \zeta)$ to improve the possibility of capturing phenomena in all probable directions. To this end, we also introduce a certain penalty for the candidate solution $\mathbf{p}$ when there are points within the $F(\ell, \zeta)$ area, which are covered by more than one nodes, and other points that are totally uncovered. Obviously, a high degree of coverage (i.e., number of nodes that cover the same point) of a certain point in the $F(\ell, \zeta)$ area is not undesirable, but, we have to guarantee first that the whole $F(\ell, \zeta)$ is covered, such that any $q \in F(\ell, \zeta)$ is covered by at least one node. If the latter objective is fulfilled, any 'additional' coverage of a certain point $q \in F(\ell, \zeta)$ will increase the robustness of the ROI coverage in terms of node failures; see Section 6.3.3.

The fitness function has as follows: a possible solution $\mathbf{p}$ is rewarded when there is some node j located at $\ell_j \in F(\ell, \zeta)$ and covers at least a point $q \in F(\ell, \zeta)$, which is not covered by another node, i.e., $q \in F(\ell_j, \delta)$. In such case, the reward equals to $\theta_q(\ell_j)$, i.e., the probability of coverage of the q point. The highest reward of unity (i.e., $\theta_q(\ell_j) \rightarrow 1$) is obtained when the *single* covered point $q \in F(\ell_j, \delta)$ is very close to the node j. Otherwise, if $q \in F(\ell, \zeta)$ is not covered by some node, then a unity penalty is assigned to the solution $\mathbf{p}$. If $q \in F(\ell, \epsilon) \setminus F(\ell, \zeta)$, i.e., q is outside of $F(\ell, \zeta)$, and it is covered by some node j then the possible solution $\mathbf{p}$ is assigned a penalty, which equals to the maximum coverage probability of nodes that cover that q, i.e., $\max_{\ell_k \in F(\ell,\epsilon) \setminus F(\ell,\zeta)} \{\theta_q(\ell_k)\}$. The later penalty refers to the situation that the location coordinates of nodes, which are located outside the $F(\ell, \zeta)$, are definitely inappropriate for minimizing the areas that are not covered by any node inside $F(\ell, \zeta)$.

The evaluation of the $\mathbf{p}$ solution derived by the $J(\mathbf{p})$ function is the cumulative reward J_R for all points $q \in F(\ell, \epsilon)$ normalized by the sum of the reward J_R and penalty

J_P values. The J_R and J_P refer to the summation of probabilities all points in $F(\ell, \zeta)$ are covered and all points in $F(\ell, \epsilon) \setminus F(\ell, \zeta)$ are not covered by any nodes. Hence, the ratio $\frac{J_R}{J_R+J_P}$ reflects a probability measure that the whole $F(\ell, \zeta)$ area is covered by the nodes. The fitness function $J(\mathbf{p})$ for a node located at ℓ, which invokes the CPSO algorithm is shown in Algorithm 2.

ALGORITHM 2: The ROI Fitness Function.

Input: Possible solution $\mathbf{p}$, area $F(\ell, \zeta)$, area $F(\ell, \epsilon)$.
Output: Degree of coverage $\frac{J_R}{J_R+J_P}$.
$J_R \leftarrow 0$, $J_P \leftarrow 0$ /*reward and penalty values*/ ;
/*for all points in ROI $F(\ell, \epsilon)$ */ ;
for $q \in F(\ell, \epsilon)$ **do**
  /*for all points in ROI $F(\ell, \zeta)$*/ ;
  if $q \in F(\ell, \zeta)$ **then**
    if $\exists \ell_k \in \mathbf{p} : \theta_q(\ell_k) \in (0, 1]$ **then**
      /*at least one node covers the point q*/ ;
      $J_R \leftarrow J_R + \theta_q(\ell_k)$;
      break ;
    else
      /*there is no node that covers the point q*/ ;
      $J_P \leftarrow J_P + 1$;
    end
  else
    /*for the rest points outside $F(\ell, \zeta)$ but inside $F(\ell, \epsilon)$*/ ;
    $R(\ell) \leftarrow F(\ell, \epsilon) \setminus F(\ell, \zeta)$;
    if $\exists \ell_k \in \mathbf{p} : \theta_q(\ell_k) \in (0, 1]$ *and* $\ell_k \in R(\ell)$ **then**
      /*there is at least one nodes that covers the point $q \in R(\ell)$*/ ;
      $J_P \leftarrow J_P + \max_{\ell_k \in \mathbf{p}}\{\theta_q(\ell_k)\}$
    end
  end
end

5.1.2. Fitness Function Complexity. In order to evaluate the $J(\mathbf{p})$ function over a candidate solution $\mathbf{p}$, we introduce the idea of discretizing the $F(\ell, \epsilon)$ area and map it into a 2-dimensional grid (lattice) of squares of dimension a, $0 < a < \epsilon$. The discretization of the area is achieved by using a Lee sphere [Golomb & Welch 1970] of radius ϵ. Specifically, the distance among the squares in the grid can be measured in terms of the Manhattan metric, that is the distance between two squares is the sum of the horizontal and vertical distances between the centers of the squares. With 'horizontal' and 'vertical' we denote the two perpendicular directions parallel to the sides of the squares in the grid. Hence, a Lee sphere of radius ϵ centered at a given square corresponding to location ℓ consists of the set of squares that lie at Manhattan distance at most ϵ from that square. This approximation facilitates the calculation of the $J(\mathbf{p})$ function where the $\theta_q(\ell)$ coverage probability involves the Euclidean distance between the points q and ℓ [de A Campello et al. 2011]. In our case, through this discretization, we map the coordinates of a point $q_{kl} = (x_{kl}, y_{kl}) \in F(\ell, \epsilon)$ with the coordinates of the center of the square at row k and column l of the grid, $k, l = 1, \ldots, \lceil \epsilon/a \rceil$. Obviously, the resolution of the square, which is determined by the discretization factor a, has impact on the computations required for $J(\mathbf{p})$ evaluation. This comes at the expense of the accuracy of the coverage probability $\theta_q(\ell_i)$ corresponding to sensor node i for each point $q \in F(\ell, \epsilon)$. The complexity of $J(\mathbf{p})$ rests on the reward/penalty assignments for

each (discrete) point $q \in F(\ell, \epsilon)$ with respect to each node location ℓ_k from the vector $\mathbf{p}$. Hence, we require $O(n\lceil \frac{\epsilon}{a} \rceil^2)$ time to evaluate the $J(\mathbf{p})$. The parameter a is a trade-off between the speed of $J(\mathbf{p})$ calculation and the accuracy of the J value. However, it should be noted that an increase in a resolution by a factor of, e.g., 10 (i.e., replacing a with $10a$) results to a decrease of the time complexity by a factor of 100, i.e., two orders of magnitude. Hence, to further significantly decrease the time complexity of the fitness function $J(\mathbf{p})$, we can simply undertake a small decrease in the resolution of the $F(\ell, \epsilon)$ area.

5.2. Stopping Criterion Analysis

In this section, we study the fitness value $J(\mathbf{p})$ of the optimal solution in light of the *required* number of iterations T^* for the CPSO algorithm to converge to the optimal value. Specifically, let T^*, with $0 < T^* \leq T$, be the iteration index at which the CPSO algorithm converges and no further improvement is observed from the $(T^* + 1)$-th iteration up to the T-th iteration for a convergence threshold ϱ, i.e.,

$$T^* = \min_{1 \leq t \leq T} \{|J_t(\mathbf{p}) - J_\tau(\mathbf{p})| \leq \varrho : t < \tau \leq T\}. \tag{15}$$

We consider the $J_t(\mathbf{p})$ value improved at t iff there is $\tau > t$ with $J_\tau(\mathbf{p}) - J_t(\mathbf{p}) > \varrho$. To this end, we provide an analysis of estimating T^* such that an improvement on the $J(\mathbf{p})$ value is observed with maximum probability $\lambda \in (0, 1)$.

The rate of convergence of a stochastic search method like the CPSO algorithm is directly dependent on the dimension of the search space $D = 2n$ and the maximum number of iterations T until convergence. The probability of generating a solution $\mathbf{p}$ in the optimality region $F_0 \subset \mathbb{R}^D$ (i.e., the probability of hitting the optimality region F_0 which is projected onto the area $F(\ell, \zeta)$) is proportional to the ratio of areas $\frac{F(\ell,\zeta)}{F(\ell,\epsilon)} = \frac{\zeta^2}{\epsilon^2}$ assuming a uniform distribution function on the search space. The authors in [Solis & Wets 1981] provide some guidelines for choosing (estimating) the number of iterations T^* required for a stochastic search algorithm to discover the optimal solution within a search space, which, in our case, the latter is projected onto the 2-dimensional disc $F(\ell, \zeta)$. Specifically, in [Solis & Wets 1981] the number of iterations T^*_λ required to reach the optimality region F_0, with at least probability $1 - \lambda$, is as follows:

$$P(\mathbf{p}_t \notin F_0) \leq \lambda, \forall t > T^*_\lambda$$

where $\mathbf{p}_t$ is the possible solution of the CPSO algorithm at the iteration index $t > 0$. If we let the probability of hitting F_0 be λ' with $0 < \lambda' \leq \frac{\zeta^2}{\epsilon^2}$ then $P(\mathbf{p}_t \notin F_0) \leq (1 - \lambda')^t$. As in [Solis & Wets 1981] choosing an integer

$$T^*_\lambda \geq \lceil \frac{\ln \lambda}{\ln(1 - \lambda')} \rceil \tag{16}$$

yields the required property, since for $t \geq T^*_\lambda$ it follows that $t \geq \ln \lambda / \ln(1 - \lambda')$ and hence $(1 - \lambda')^t \geq \lambda$. If, in our case, we use $\lambda' = \frac{\zeta^2}{\epsilon^2}$, then this implies that the number of iterations required to reach F_0 with probability $1 - \lambda$ is

$$T^*_\lambda \geq \frac{\ln \lambda}{\ln(1 - (\frac{\zeta}{\epsilon})^2)}. \tag{17}$$

Indicatively, if $\zeta = (0.2\epsilon, 0.7\epsilon)$ and probability $1 - \lambda = 0.85$ then $T^*_{0.15} = (47, 3)$ iterations, respectively; for $1 - \lambda = 0.99$ then $T^*_{0.01} = (113, 7)$ iterations, respectively; for $1 - \lambda = 0.999$ then $T^*_{0.001} = (170, 11)$ iterations, respectively. Hence, as will be shown below, we choose $T = 100$ iterations with a minimum value of $T^* = 10$ to reach the optimality region with a minimum probability $1 - \lambda = 0.99$.

5.2.1. CPSO Performance Example. We proceed with a running example of the CPSO algorithm demonstrate its performance. The parameters of the CPSO algorithm that are adopted for the ROI coverage Problem 1 are summarized in Table I. Here, we study the impact of the parameters: n (number of nodes; particle dimension), m (number of particles; population size), T (maximum number of iterations), and ζ (radius of the coverage ROI) on the behavior of the CPSO algorithm in our Problem 1.

Table I. The parameters for CPSO in the $F(\ell, \zeta)$ coverage problem.

Parameter	**Description**	**Value/Range**
n	Number of nodes within $F(\ell, \epsilon)$. It is the $D = 2n$ particle dimension.	$\{10, \ldots, 60\}$
m	Population size (number of particles).	$\{10, \ldots, 60\}$
ζ	Radius of the coverage ROI $F(\ell, \zeta)$. It is a function of the communication range ϵ.	$[0.2\epsilon, 0.7\epsilon]$
ϵ	Communication range.	50m
δ	Sensing range.	3m
T	Maximum number of iterations.	100
ϱ	Convergence threshold.	10^{-5}
a	Discretization factor of the $F(\ell, \epsilon)$ area.	0.5m
$1 - \lambda$	Probability of hitting the optimality region.	0.99

Figure 2 (left) illustrates $J(\mathbf{p})$ per iteration $t = 1$ up to $t = 20$ for different number of nodes n, $\delta = 6$m, $\epsilon = 50$m, and $\zeta = 20$m = 0.4ϵ. We also set the discretization factor of the $F(\ell, \epsilon)$ area $a = 0.5$m, i.e., the square that encloses that area contains $\lceil \epsilon/a \rceil^2 = 10^4$ squares of dimension 0.5m. The CPSO algorithm with $m = 20$ particles converges after $t = 10$ iterations for all n, i.e., $T^* = 10$. We can obtain over 95% coverage of $F(\ell, \zeta)$ for $n \geq 15$ nodes. Figure 2(right) shows the convergence of the $J(\mathbf{p})$ value for different values of ζ ($0.2\epsilon \leq \zeta \leq 0.7\epsilon$) with population size $m = 20$. We observe that the number of nodes that are needed for covering the 95% of $F(\ell, \zeta)$ is greater than 20 with a radius between 0.5ϵ and 0.6ϵ. Obviously, once the number of nodes that are needed to cover the $F(\ell, \zeta)$ is not sufficient, then there might be some regions within $F(\ell, \zeta)$ in which a possible phenomenon will not be detected. We experiment with different values of m and n, having $\zeta = 0.35\epsilon$ and the corresponding convergence $J(\mathbf{p})$ value is shown in Table II (similar values are obtained for $\zeta > 0.35\epsilon$). From these results we can observe that, for a given ζ, the corresponding ROI $F(\ell, \zeta)$ can be 95% covered having more than 20 nodes. Moreover, a better solution is obtained as the number of population increases. However, as $m > 30$ then there is no significant improvement on the convergence of $J(\mathbf{p})$. Accordingly, we can obtain a degree of coverage 95% with $m = 20$.

Table II. The convergence $J(\mathbf{p})$ value with $\zeta = 0.35\epsilon$.

	Population size m					
# nodes n	**10**	**20**	**30**	**40**	**50**	**60**
10	0.77	0.87	0.88	0.90	0.90	0.91
20	0.96	0.97	0.98	0.99	0.99	1.0
40	0.98	0.99	0.98	1.0	1.0	1.0
60	0.99	1.0	0.99	1.0	1.0	1.0

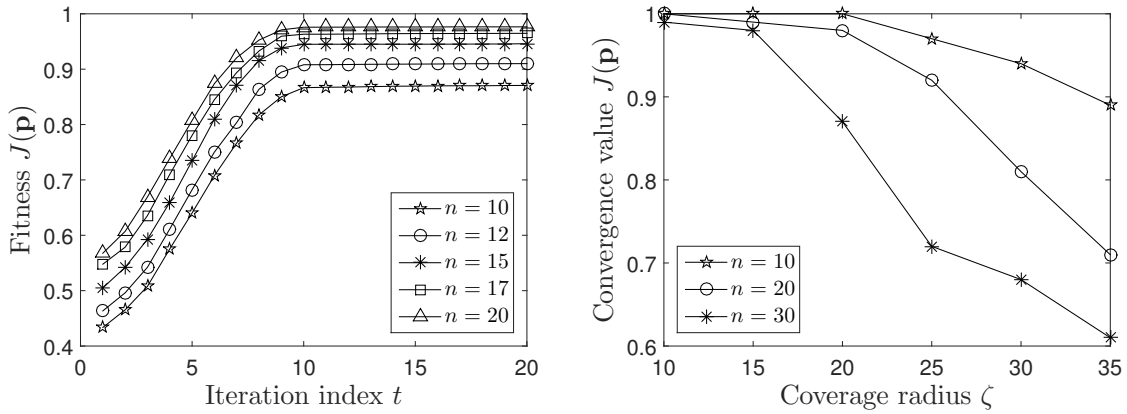

Fig. 2. (Left) The $J(\mathbf{p})$ value per iteration t for different n (population size $m = 20$); (right) the $J(\mathbf{p})$ convergence value vs. ζ for different n (population size $m = 20$).

5.3. Computational Complexity

The computational complexity of the CPSO in Algorithm 1 with the proposed fitness function in Algorithm 2 for the optimization Problem 1 depends on (i) the number of particles m, (ii) the number of iterations T, with $T > T^*$, and (iii) the time complexity of the fitness function $J(\mathbf{p})$. Specifically, for a given discretization factor a of a specific $F(\ell, \epsilon)$ area, a vanguard node requires $O(mTn\lceil \frac{\epsilon}{a} \rceil^2)$ time. The trade-off between computational time and probability of coverage is (i) the factor a and (ii) the required number of iterations given a certain probability of hitting the optimality region $1 - \lambda$. The higher the a value, the lower the number of discrete points required for evaluating the coverage probability of the $F(\ell, \epsilon)$ area. Moreover, the higher the λ the lower the number of required iterations T^* becomes. This however comes at the expense of a suboptimal solution.

5.3.1. Computational Complexity Example. We first study the impact of the number of neighbouring nodes n, the number of particles m and the coverage radius ζ on the time complexity of the ROI coverage algorithm. Starting at iteration $t = 1$, we measure the overall time in milliseconds in which the CPSO algorithm converges at the T^*-th iteration, i.e., the minimum required time for the CPSO algorithm to converge with respect to certain m, n, and ζ values with the default a and λ values in Table I. The absolute time (in msec) quantifies the time needed for a vanguard node (e.g., mobile robot) to reach an optimal solution for the ROI coverage in real situations, when the phenomena are dynamic (progressing over time)[3]. Moreover, we also study the required number of iterations T^* against the discretization factor a for different values of the number of neighboring nodes and particles in the CPSO algorithm.

Figure 3(left) illustrates the minimum time T^* (in msec) for the CPSO algorithm to converge against m for different n values and $\zeta = 0.35\epsilon$. We can observe that for number of nodes $n = 10$, the CPSO algorithm requires a high number of iterations, thus, high computational effort, to converge, however, with low convergence values, as shown in Table II. Moreover, given a sufficient number of nodes (i.e., corresponding to 95% of coverage with $n \geq 20$ and $\zeta = 0.35\epsilon$), the execution time of the CPSO algorithm

[3]The memory and CPU characteristics of a robot for measuring this time refer to a Swarm-Bot [Mondada et al. 2004] with 400 MHz XScale CPU board, 64 MB RAM, and 32 MB of flash memory. Details regarding the hardware and simulation of the swarm-bot can also be found at http://www.swarm-bots.org.

is quite low. Furthermore, in order to obtain a better solution with a high degree of coverage (as shown in Table II), then a higher m value is required. However, the computational time of the CPSO algorithm to converge does not significantly increase as the m value increases. Hence, with a value of $m = 20$ we can obtain a minimum 95% coverage within a relatively low execution time of the CPSO algorithm.

Figure 3(right) shows the T^* against ζ for different values of m and $n = 30$. In Figure 3(left) we show how the execution time of the CPSO algorithm depends on the coverage radius ζ as a function of the communication range ϵ. Specifically, given a certain number of nodes ($n = 30$), a low value of ζ indicates that the CPSO algorithm has to optimally locate nodes in a very close area w.r.t. communication range. In this case, the CPSO algorithm attempts to find the best possible solution, since the number of nodes are sufficient to optimally cover the required ROI. Hence, we could adopt a low value for the population size in order to achieve a low execution time of the CPSO algorithm. On the other hand, as ζ approaches the communication range ϵ, the CPSO algorithm converges very fast to a solution, but, this solution is sub-optimal. That is because, the number of nodes are not sufficient enough to cover the ROI $F(\ell, \zeta \to \epsilon)$, which yields a 90% coverage as shown in Figure 2(b) with $n = 30$. In this case, any attempt to increase the population size will not result to a better solution. Hence, given a number of nodes n and a radius ζ, we can choose a population size m in order to obtain a relatively high degree of coverage, i.e., over 95%. Overall, a value of $m = 20$ ensures a relatively low CPSO execution time with 95% coverage having a minimum number of nodes $n = 20$.

We further study the impact of the discretization factor a on the ROI coverage algorithm. Figure 4(left) shows the required number of iterations T^* against a values such that $\frac{\epsilon}{a} \in \{16.67, 25, 50, 100, 500\}$, for coverage radius $\zeta = 0.35\epsilon$, $\epsilon = 50$m, $n = 30$ and with different values of m. Note that a value of $a > \delta$, i.e., greater than the sensing range would result to a rather inaccurate estimation of the coverage probability of each point within the $F(\ell, \epsilon)$ area since the sensing region of a node is smaller than the square which represents the discrete point in the Lee sphere of radius ϵ. Hence, we restrict a in the interval $(0, \delta]$. From Figure 4(left) one can observe that even for a high degree of discretization, e.g., $a = 0.1$m with a radius $\epsilon = 50$m, the maximum number of the T^* value is below 70 iterations. Evidently, the higher the number of particles, the more candidate solutions have to be examined and explore the search space, thus, increasing the time complexity. However, as discussed above a number of particles $m = 20$ ensures low CPSO execution time with relatively high degree of coverage. This denotes a required number of $T^* = 40$ given a high resolution of the Lee sphere of radius ϵ. It is also interesting to show the required number of iterations T^* against the discretization ratio $\frac{\epsilon}{a}$ for different number of (neighbouring) nodes n having $m = 20$ particles and $\zeta = 0.35\epsilon$. For relatively low number of nodes n, the CPSO algorithm requires a relatively high number of iterations since, given the required coverage radius of ζ, the coverage degree of the ROI is not achievable. That is, more iterations are needed for those nodes to optimally cover the area under consideration. When the number of nodes is relatively high, e.g., $n > 20$, then an optimal solution is obtained within a relatively low number of required iterations since more nodes cover the required area reflecting a high degree of coverage. As discussed above, a minimum number of $n = 20$ is required to achieve at least a degree of coverage close to 95%. Given this, the discretization ratio $\frac{\epsilon}{a}$ does not significantly impact the time complexity of the CSPO in terms of the required number of iterations T^*. This indicates that we could avoid further calculations of the CPSO algorithm when the number of nodes is relatively high, thus, saving energy on the node that executes the CPSO algorithm. A discussion of the minimum number of nodes $n_{\min}$ is also provided in Section 6.2.2.

Remark 5.1. It is worth noting that we need only to know the locations of the neighbours within $F(\ell, \epsilon)$ since the ROI coverage algorithm is executed on a node located at ℓ (with communication range ϵ). This reflects the distributed nature of the proposed approach in the phenomena localization process.

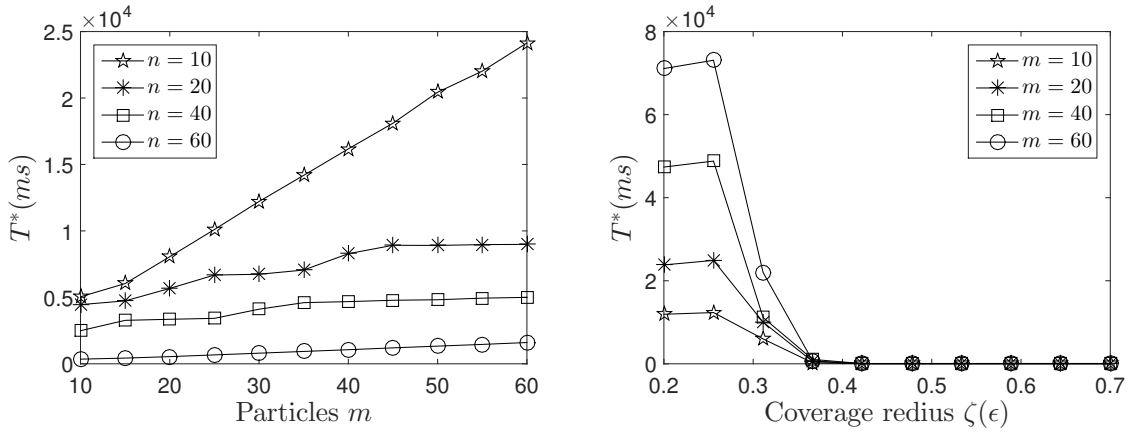

Fig. 3. (Left) Minimum time T^* against population size m for different number of nodes n with $\zeta = 0.35\epsilon$; (right) Minimum time T^* against coverage radius ζ as a function of ϵ for different population size m with $n = 30$.

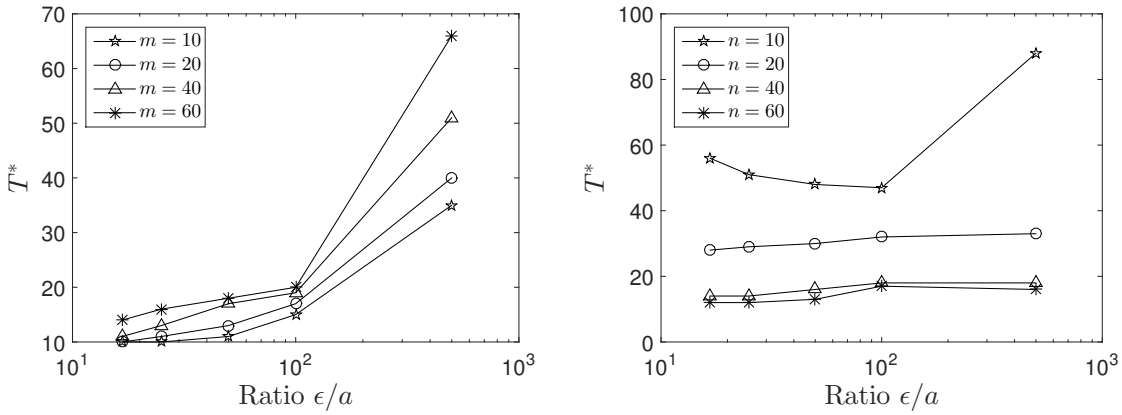

Fig. 4. (Left) Required number of iterations T^* against ratio ϵ/a for different population size m and number of nodes $n = 30$ with $\zeta = 0.35\epsilon$; (right) Required number of iterations T^* against ratio ϵ/a for different number of nodes n with population size $m = 20$ and $\zeta = 0.35\epsilon$.

6. THE DISTRIBUTED PHENOMENA LOCALIZATION PROCESS

6.1. Overview

This section discusses the proposed distributed process for localization of phenomena occurring in a ROI F. The distributed nature of the process is based on the fact that it is independently executed on the *vanguard* nodes. A vanguard, which better detects a phenomenon compared to its neighbours, is temporally elected by its neighbourhood

as discussed later in Section 6.3. Through such algorithm nodes are self-organized, i.e., can autonomously relocate to the areas close to phenomena, i.e., POIs. The basic subprocesses of the algorithm are the gathering and the election process. In the gathering process, Section 6.2, the vanguards group together neighbours through optimal relocation messages, referred to as **Relocation Directives** (RD), for further inspection of a certain region. In the election process, Section 6.3, a neighbourhood elects a unique vanguard as the leader for the *next* gathering process. Once nodes stop relocating, they reach at the POIs, until the phenomena are eliminated or another relocation message 'awakes' them for initiating a new detection process. The algorithm results to autonomous merging and expansion of neighbourhoods with the corresponding minimum number of vanguards.

The entire process deals with:

— How a vanguard is elected.
— How neighbours receive RDs by a vanguard instructing them where to relocate and where to sense.
— How vanguards, with quite similar magnitude values, merge/expand their neighbourhoods while targeting to the same and/or different POIs.

The proposed process in Algorithm 3 shows all subprocesses and methods. Specifically, the vanguards $\mathcal{V} \subseteq \mathcal{N}$ are elected through the distributed election process, **election**($\mathcal{N}$), which is introduced in Section 6.3. Then, each vanguard $i \in \mathcal{V}$ executes (locally) the CPSO algorithm, which is referred to as **coverage**($F(\ell_i, \zeta)$), introduced in Section 5. Through such coverage subprocess each vanguard i obtains the optimal locations $\mathbf{p}_{\mathbf{g}_i}$. Based on the $\mathbf{p}_{\mathbf{g}_i}$, the vanguard i sends the RDs, let us define that by **send**($j, \langle (x_j^*, y_j^*), \mu_{\phi,i} \rangle$), to each neighbouring node $j \in \mathcal{N}_i$ to further sense the area around the vanguard as shown in Figure 1 and Figure 6. Consequently, each neighbouring node $j \in \mathcal{N}_i$ adjusts locally its next movement, **update**(ℓ_i), with respect to two proposed policies, which are introduced in Section 6.2.1; see BRP using Eq(18) or WRP using Eq(19). The **sense**(i, Ψ) subprocess refers to the update of the magnitude levels for node i with respect to the current observable phenomena Ψ. A node (vanguard) stops relocating (gathering), **stop**(i), once it oscillates close to its current location given a relatively small oscillation threshold $\theta > 0$. In the remainder of this section, we elaborate on each subprocess of the entire phenomena localization process and report on the expected energy cost for each node and the entire MSN.

6.2. The Gathering Process

Let a set $\mathcal{N}$ of randomly distributed nodes on F at time $t = 0$. Consider at time instance $t > 0$ some nodes $\mathcal{V}(t) \subset \mathcal{N}$ detect certain phenomena with non-zero magnitude value $\mu_{\phi,i}(t), i \in \mathcal{V}(t)$. Such phenomena Φ, which are detected by these nodes in $\mathcal{V}(t)$, are the 'observable phenomena' $\Psi(t) \subseteq \Phi$, i.e., $\Psi(t) = \{\phi \in \Phi : \mu_{\phi,i}(t) > 0, i \in \mathcal{V}(t)\}$. The nodes from the $\mathcal{V}(t)$ set, hereinafter, are referred to as the **vanguards**. Vanguards are either the first nodes that detect phenomena (first responders) or better detect a phenomenon compared to their neighbours as shown in Figure 6. A vanguard i, located at ℓ_i, stimulates a collaborative scan of the nearby area within its sensing region $F(\ell_i, \delta)$ in order to better perceive the evolving phenomena. Hence, it gathers its neighbours to cover the broader area of $F(\ell_i, \delta)$ as shown in Figure 1.

A vanguard $i \in \mathcal{V}(t)$ detects a phenomenon ϕ with the highest magnitude value among its neighbours, i.e.,

$$\mu_{\phi,i}(t) = \max_{j \in \mathcal{N}_i(t)} \mu_{\phi,j}(t),$$

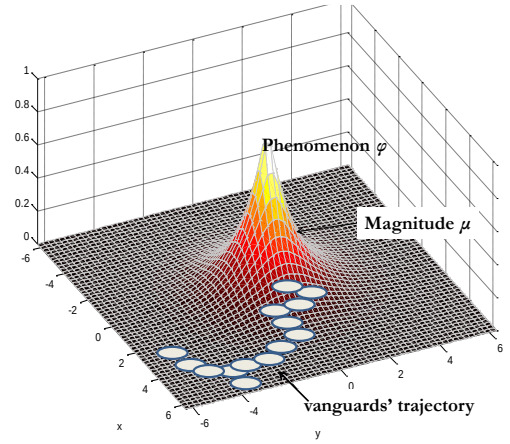

Fig. 5. Illustration of the vanguards' trajectory towards a phenomenon; the vertical axis (z-axis) refers to the magnitude value (e.g., sensed radiation) of the phenomenon ϕ.

ALGORITHM 3: The Phenomena Localization Process.

Input: Set of nodes $\mathcal{N}$.
while *TRUE* **do**
  /*Distributed vanguard election subprocess*/ ;
  $\mathcal{V} \leftarrow$ **election**($\mathcal{N}$);
  for *each vanguard* $i \in \mathcal{V}$ **do**
    /*Distributed coverage subprocess; local execution of CPSO*/ ;
    $\mathbf{p}_{\mathbf{g}_i} \leftarrow$ **coverage**($F(\ell_i, \zeta)$);
    for *each neighbouring node* $j \in \mathcal{N}_i$ **do**
      /*vanguard i sends relocation directives*/;
      send($j, \langle (x_j^*, y_j^*), \mu_{\phi,i} \rangle$);
    end
  end
  /*All neighbouring nodes from all swarms in the network adjusts their location*/;
  for *each node* $j \in \cup_{i \in \mathcal{V}} \{\mathcal{N}_i\}$ **do**
    update(ℓ_j);
  end
  /*All nodes in the network updates their magnitude levels corresponding to certain phenomena*/;
  $\Psi = \{\phi \in \Phi : \mu_{\phi,i} > 0, i \in \mathcal{V}\}$;
  for *each node* $i \in \mathcal{N}$ **do**
    sense(i, Ψ);
  end
  /*Check if a node stops relocating/gathering in the field*/;
  for *each node* $i \in \mathcal{N}$ **do**
    if $|\Delta \ell_i| < \theta$ **then**
      stop(i);
    end
  end
end

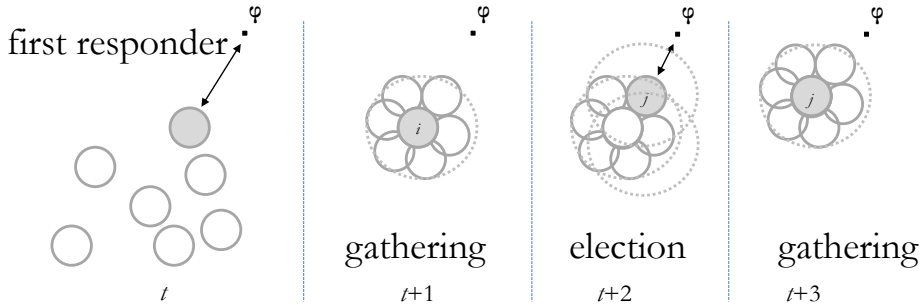

Fig. 6. From left to right: the gray-colored node i, which refers to the vanguard, is the first responder at time/step t. Such node gathers its neighbouring nodes around it for further exploring the area at step $t+1$. Some node j then senses a higher magnitude value of the phenomenon ϕ than the other nodes, thus, becoming a new vanguard after a local election process at step $t+2$. Consequently, another gathering process is performed at step $t+3$ relocating the neighbouring nodes of the newly vanguard j around it and so no.

with $\phi \in \Psi(t)$. The strategy of vanguard i is to gather its neighbours $\mathcal{N}_i(t)$ around $F(\ell_i, \zeta) \supset F(\ell_i, \delta)$ with $\epsilon > \zeta > \delta$. To this end, each vanguard independently executes the CPSO algorithm for $F(\ell_i, \zeta)$ with $n = |\mathcal{N}_i(t)|$ nodes. The only knowledge for evaluating the $J(\mathbf{p})$ function of the CPSO algorithm is the locations of the neighbours of the vanguard. Hence, each vanguard obtains the n optimal locations, i.e., the solution $\mathbf{p_g}$ in Eq(13), for its neighbours. Such locations are transmitted by the vanguard to the neighbours through the relocation directives, RDs.

Once neighbours relocate to optimally cover $F(\ell_i, \zeta)$ then some neighbour $j \in \mathcal{N}_i(t)$ is likely to detect the same (or another) phenomenon ϕ at time instance $t+1$ with higher magnitude value $\mu_{j,\phi}(t+1)$ than that of the current vanguard ($\mu_{i,\phi}(t)$). In this case, at time $t+1$, node j is elected to become a vanguard, as will be discussed in Section 6.3, and repeats the same strategy to its neighbours as shown in Figure 6 and 7. Through this 'domino-like' process, the nodes exchange the leadership roles represented as vanguards and lead their neighbours to areas where they detect phenomena with high magnitude values, as illustrated in Figure 6 and 7. If, at time t, none of the neighbours detects a phenomenon with higher magnitude than that of the vanguard at $t-1$ then the vanguard holds the leadership at t. This indicates that the vanguard locates very close to the POI.

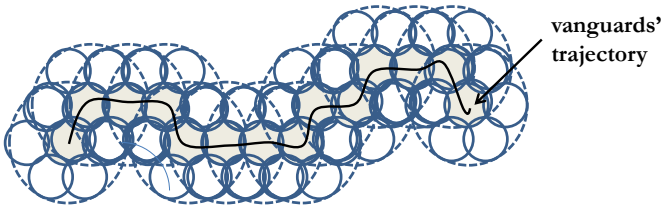

Fig. 7. The 'domino-like' process; the nodes exchange leadership roles and lead (through relocation directives) their neighbours to further explore areas where they detect phenomena with higher magnitude values.

6.2.1. The Relocation Directive. All vanguards at time t execute, in parallel, the $F(\ell_i, \zeta)$ coverage process with $|\mathcal{N}_i(t)|$ nodes, $i \in \mathcal{V}(t)$. Then, the vanguard i sends the RDs to its

neighbours $j \in \mathcal{N}_i(t)$. A RD from vanguard i to neighbour j is the tuple

$$\langle (x_j^*, y_j^*), \mu_{\phi,i}(t) \rangle,$$

where $\ell_j^* = (x_j^*, y_j^*) \in \mathbf{p_g}$ is the optimal location for neighbour j and $\mu_{\phi,i}(t)$ is the magnitude value evaluated on vanguard i at time t. The neighbour j, then, adopts one of the two our proposed policies for the received RD:

— The node j directs at the next time instance $t+1$ towards to the indicated location $(x_j^*, y_j^*) \in F(\ell_i, \zeta)$ with constant speed, i.e.,

$$\ell_j(t+1) = \ell_j^* \tag{18}$$

We refer to such policy as **blind relocation policy** (BRP).

— The node j adjusts its next movement based on the $\mu_{\omega,j}(t)$ value, which has been evaluated by node j (assuming that node j detects a phenomenon ω; otherwise $\mu_{\omega,j}(t) = 0$), and the $\mu_{\phi,i}(t)$ value. That is, node j moves on the direction toward the optimal location ℓ_j^* but stops at location $\ell_j(t+1)$ such that

$$\ell_j(t+1) = \ell_j(t) + f(\mu_{\phi,i}(t), \mu_{\omega,j}(t))(\ell_j^* - \ell_j(t)) \tag{19}$$

The function $f(\mu_{\phi,i}(t), \mu_{\omega,j}(t)) \in [0, 1)$ weights the direction vector of node j toward to ℓ_j^*. We refer to such policy as **weighted relocation policy** (WRP).

In the WRP, the $f(\mu_{\phi,i}(t), \mu_{\omega,j}(t))$ value depends on the percentage difference $\frac{|\mu_{\phi,i}(t) - \mu_{\omega,j}(t)|}{\mu_{\phi,i}(t)}$ of the magnitude that node i and node j evaluate for the phenomena ϕ and ω, respectively, and their distance $\|\ell_i - \ell_j\|$ with respect to the sensing radius δ. If node j does not detect any phenomenon (i.e., $\mu_{\omega,j}(t) = 0$) then we adopt a unity weight and, thus, the BRP (i.e., $\ell_j(t+1) = \ell_j^*$).

It is worth mentioning that, in our solution, there is no assumption that node i and node j detect the same phenomenon (source) or not. Therefore, there is no information on whether nodes i and j detect the same or different phenomenon. Even with the absence of such important information the algorithm directs nodes close to the POIs. Moreover, the ϕ and ω phenomena could be either the same or different. Specifically we distinguish the following cases:

— **Case A**: $\mu_{\omega,j} \ll \mu_{\phi,i}$, i.e., node j's $\mu_{\omega,j}$ is relatively smaller than $\mu_{\phi,i}$ of vanguard i. In this case, node j should follow vanguard i, irrespective of the fact that phenomenon ω is the same as the phenomenon ϕ, which the latter is sensed by the vanguard. Hence, the node j should adopt the RD dictated by its vanguard in light of further examining the localization of the phenomenon ϕ (in favor of vanguard). This is denoted with a weighted value close to unity, which results to the movement of node j close to the indicated optimal location ($\| \ell_j(t+1) - \ell_j^* \| \simeq 0$).

— **Case B**: $\mu_{\omega,j} \simeq \mu_{\phi,i}$, i.e., the $\mu_{\omega,j}$ of node j is relatively close to $\mu_{\phi,i}$ of vanguard i. In this case we further consider the following cases:

  — **Case B.1: (Merge)**. The fact that $\mu_{\omega,j} \simeq \mu_{\phi,i}$ might correspond to the case where node j and vanguard i are not far away from each other with respect to their sensing radius 2δ. In this case, probably, node j senses the same phenomenon with its vanguard i. That is, if their distance $\|\ell_i - \ell_j\|$ is less than 2δ, then the node j should follow its vanguard, thus, the bias weight should be close to unity. That is because, the node j has approximately sensed the same magnitude with that of the vanguard and is close to the vanguard. Hence, they have both probably sensed the same phenomenon and it is meaningful for node j to further examine the broader area than simply relocate close to vanguard i's area. In the case that $k > 1$ vanguards detect the same phenomenon then, at the next election process,

the node i with the highest magnitude value becomes the vanguard. Hence, certain nodes from the neighbourhoods $\mathcal{N}_k$ group together to one neighbourhood $\mathcal{N}_i$ with the node i as vanguard; see Figure 8.

— **Case B.2: (Split)**. The fact that $\mu_{\omega,j} \simeq \mu_{\phi,i}$ might correspond to the case where node j and vanguard i are far away from each other with respect to their sensing radius δ. Specifically, if the distance $\|\ell_i - \ell_j\|$ between the node j and the vanguard i is greater than 2δ, then probably the phenomenon ω is different with the ϕ. In that case, it would be preferable to leave node j to further investigate and localize the phenomenon ω than following vanguard i. That is, node j should avoid follow the RD of its vanguard in favor of its phenomenon investigation. In this case, the bias relocation weight depends on the slight difference $\mu_{\omega,j} - \mu_{\phi,i}$ and on the distance $\|\ell_i - \ell_j\|$. If this distance is greater than 2δ, then node j splits from the vanguard's swarm and helps in splitting $\mathcal{N}_j$ into two (or more) neighbourhoods with different vanguards and then expanding to different neighbourhoods; see Figure 9. In this case the bias weight is very close to zero.

Overall, the rules for the RD for the WRP are as follow:

— If $\mu_{\omega,j} \ll \mu_{\phi,i}$ irrespective of the fact that $\phi = \omega$ or $\phi \neq \omega$, then the bias weighting function $f(\mu_{\phi,i}(t), \mu_{\omega,j}(t)) = 1$.
— If $\mu_{\omega,j} \simeq \mu_{\phi,i}$ and $\|\ell_i - \ell_j\| > 2\delta$, then the bias weighting function $f(\mu_{\phi,i}(t), \mu_{\omega,j}(t)) < 1$ since, probably, $\phi \neq \omega$.
— If $\mu_{\omega,j} \simeq \mu_{\phi,i}$ and $\|\ell_i - \ell_j\| \leq 2\delta$, then the bias weighting function $f(\mu_{\phi,i}(t), \mu_{\omega,j}(t)) = 1$ since, probably, $\phi = \omega$.

Based on the above rules, we define the bias weighting function $f : [0, \mu_{\max}] \times [0, \mu_{\max}] \to [0, 1]$ as follows:

$$f(\mu_{\phi,i}, \mu_{\omega,j}) = \begin{cases} 1 - e^{-\left(\frac{|\mu_{\phi,i} - \mu_{\omega,j}|}{\mu_{\phi,i}} \frac{1}{\gamma}\right)^2}, \text{if } \|\ell_i - \ell_j\| > 2\delta \\ 1, \text{otherwise.} \end{cases} \tag{20}$$

where $\gamma \in (0, \infty)$ is a parameter for balancing the importance of the percentage difference of the magnitude values. It should be noted that through $f(\mu_{\phi,i}, \mu_{\omega,j})$ with $\|\ell_i - \ell_j\| > 2\delta$, the coverage of $F(\ell_i, \zeta)$ is sub-optimal with respect to the CPSO algorithm. That is because, some members of the vanguard i neighborhood $\mathcal{N}_i$, with respect to their magnitude values, do not strictly follow the RDs of their vanguard i. This is attributed to the fact that for these members a reception of relatively high magnitude values, which are close to that of their vanguard, refers to other possible phenomena in the broader ROI. Hence, instead of blindly following the RDs of their vanguard (i.e., relocate to the dictated optimal locations), they start off exploring their areas for further identifying new phenomena different with that their vanguard i had identified. As we elaborate in Section 6.2.2 there are constraints on the number of members that split the current neighborhood of a vanguard for exploring other areas different than those indicated by the vanguard i. As $\gamma \to \infty$ the value of f tends to zero; node j does not strictly follow the RD received by its vanguard. In the case where $\gamma \to 0$, f tends to unity ($f \to 1$), the WRP behaves as the BRP. The latter case is observed whether neighbours have zero or very low magnitude values relatively to their vanguard. Once the magnitude values of the neighbours are relatively close to that of their vanguard then there is no obligation for some of them to follow the vanguard's RDs, as explained before. Note also that the value of the distance $\|\ell_i - \ell_j\|$ is know to the vanguard i, $\forall \ell_j$, since vanguard i uses these locations for the CPSO algorithm.

Once nodes are getting closer to a POI through the RDs the weighting function tends to zero and their traveled distance decreases. The nodes gradually reduce the trav-

eled distance until becoming static and close to the actual POI and the corresponding vanguard is located very close to that POI (see Figure 5). Furthermore, through the weighting function nodes with relatively equal magnitude values do not move close to each other. Hence, they are independent to further examine the area.

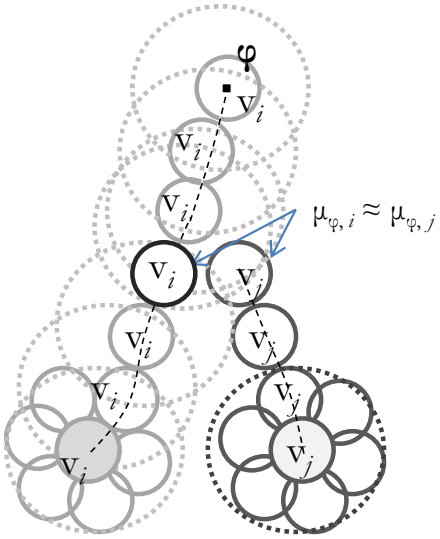

Fig. 8. Merge: the two vanguards i and j of two different *swarms* detect the same phenomenon ϕ with their local magnitude values $\mu_{\phi,i}$ and $\mu_{\phi,j}$, respectively, are very close each other. Then, in the next election process a new vanguard is elected from the merging of the two swarms.

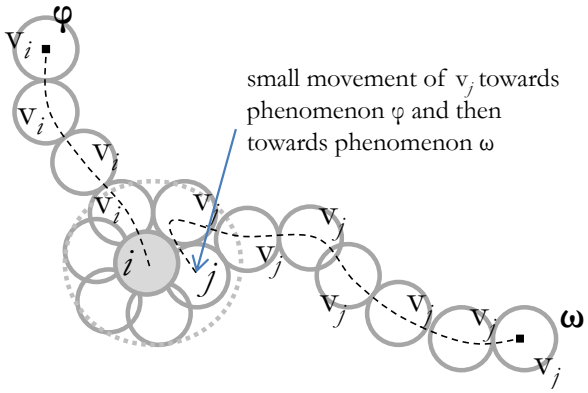

Fig. 9. Split: the vanguard i and its member node j receive two magnitude values, which are very close to each other but correspond to two different incidents ϕ and ω, respectively. Then, the member node j starts exploring more its own area instead of following the RD from its vanguard i, by splitting the *swarm* into two parts.

Remark 6.1. If node j receives RDs by more than one vanguards then it follows the RD sent by the vanguard $i \in \mathcal{V}(t)$ with the maximum μ_i and $j \in N_i$. In addition, if node j receives RDs by more than one vanguards with relatively similar magnitudes, then node j adopts one of the received RDs at random. The term similar magnitudes implies that for each pair of vanguards (κ, κ') with $\kappa, \kappa' \in \mathcal{V}(t)$ and $j \in N_\kappa$ and $j \in N_{\kappa'}$ we have that the difference $|\mu_\kappa - \mu_{\kappa'}|$ is very small. As will show in the election process in Section 6.3, a node is assigned only to one vanguard.

6.2.2. Constraints of the neighborhood split. As mentioned above, a neighborhood split is initiated by a node $j \in \mathcal{N}_i$, with $n = |\mathcal{N}_i|$, which does not follow the RDs of its vanguard node i because of the fact that node j and node i measure quite similar magnitude values. The rate, at which neighborhood splits occur, depends on the relative position of nodes and their vanguard with respect to the phenomena POIs. When some nodes are quite close to their vanguards and there are multiple phenomena whose POIs are close to those nodes, then there is a high likelihood that those nodes will proceed with a neighborhood split. In that case, there has to be a constraint on the number of nodes $n' = |\mathcal{N}'|$, with $\mathcal{N}' \subset \mathcal{N}_i$, that corresponds to the split of the neighborhood $\mathcal{N}_i$, since the remaining $n - n'$ nodes in that neighbor ($\mathcal{N}_i \setminus \mathcal{N}'$) have to continuously cover the ROI $F(\ell_i, \zeta)$. This means that, a neighboring node $j \in \mathcal{N}_i$, which is about to leave the vanguard's neighborhood, has to gain a 'permission' to proceed with this split by node i (its vanguard). This information should be determined by the vanguard, which refers to finding the minimum number of nodes $n_{\min} = n - n'$ that are needed to cover the ROI $F(\ell_i, \zeta)$. This resembles to the well-known Art Gallery problem [Marengoni et al. 1996], which aims at determining the number of observers (nodes, in our case) necessary to cover an art gallery room (the ROI $F(\ell_i, \zeta)$, in our case) such that every point is covered by at least one observer. Nevertheless, covering a bounded region with a minimum number of nodes is a NP-complete problem [Paterson et al. 1981]. However, there exist various approximate solutions that run in polynomial time and have a bounded error [Cook et al. 2001, Kar et al. 2003]. Such methods could be adopted by the vanguard in order to determine $n_{\min}$ and, thus, grant permission to node(s) for a neighborhood split. For instance, since ROI $F(\ell, \zeta)$ is convex, then based on [Kar et al. 2003] and adopting the corresponding δ-strip coverage method, the number of nodes $n_{\min}$ needed to provide connected coverage to $F(\ell_i, \zeta)$ is equal/less than $2.693(1+4.486\frac{\delta}{\zeta})n^*_{\min}$, where $n^*_{\min}$ is the optimal solution; see also [Cook et al. 2001]. In this paper, for a given ratio $\frac{\zeta}{\epsilon}$, based on the experimental results in Table II, the vanguard can determine the minimum value of the remaining nodes $n_{\min}$ within its neighborhood in order to achieve a ROI coverage of 95%. In the case where $\frac{\zeta}{\epsilon} = 0.35$, we obtain $n_{\min} = 20$. Hence, a node j has the permission to leave a vanguard's neighborhood once the number of nodes that will remain in the neighborhood is at least $n_{\min}$. This permission can be requested by a node, as a message to the vanguard, after the latter has transmitted the RD to it. Our future research agenda includes the study of a vanguard to determine the sufficient number of nodes that are needed to cover the current ROI and to manage neighborhood splits.

6.3. The Election Process

6.3.1. Requirements of the election process. The election process concerns a node to become dynamically a vanguard that it experiences the highest magnitude value of a sensed phenomenon among its neighbors. The proposed distributed election process is mapped to node clustering, with formed cluster heads (i.e., vanguards) from the set $\mathcal{N}$ of the nodes in the MSN. The aim is to notify the remaining nodes about their vanguard avoiding redundant message dissemination. The remaining nodes (members) then await for RDs from their vanguards to better identify and localize the possible

phenomena in the ROI. The election process in re-evaluated once all nodes have been relocated to the specified optimal locations for better coverage, thus, triggering a new election process which results to new vanguards appointments and so no; see Figure 6.

The primary objectives of the election process are: (i) appointment of a subset of nodes as vanguards responsible for determining and disseminating the RDs to their member nodes and (ii) dynamically changing the vanguard appointment to nodes. Evidently, this prolongs the MSN lifetime by changing vanguard appointments and, thus, balancing energy consumption of the CPSO coverage algorithm and transmission of the RDs to the members; (iii) terminating the election process within a constant number of iterations (exchanged messages).

It is worth noting that we do not make any assumption about the spatial distribution of nodes on the ROI. We only assume that nodes can measure their remaining energy and the magnitude value due to the phenomenon occurrence. In our model, every node can act as both a vanguard and a member, which motivates and requires the need for efficient election algorithms to elect vanguards. A node may fail if its energy resource is depleted, which increases the need for rotating the vanguard leadership among neighbouring nodes for load balancing. We assume that node failures are caused by energy depletion and discuss measures to improve robustness due to unexpected node failures in harsh environments e.g., volcanic areas or military fields, as discussed in Section 6.3.3.

A baseline solution for the election process involves nodes exchanging / flooding their magnitude values and residual energy budgets to all neighbouring nodes. Hence, the node with the highest energy budget and the highest magnitude value is elected to become the vanguard within its connectivity area. However, this solution requires a significant number of messages exchanged among nodes. Moreover, since the election process is re-initiated after the optimal relocation of the nodes, then a high energy budget is required for that type of communication. In our case, we introduce the following requirements for the election process:

— The vanguard election is completely distributed. Each node independently makes its decisions based only on local information, i.e., magnitude value of sensed phenomena and remaining energy.
— The election terminates within a fixed number of iterations regardless of the MSN diameter.
— At the end of the election process, each node is either a vanguard or a member that awaits to receive RDs from exactly one vanguard.
— The election process should be efficient in terms of processing complexity and message exchange.
— The vanguards have the highest average residual energy compared to member nodes and have relatively the highest magnitude values of the sensed phenomena in the considered ROI.
— The vanguard must be able to directly communicate with its members in a single hop.

6.3.2. Cluster-based election process. There are certain election algorithms which can be adopted in a neighbourhood of mobile nodes. In our case, neighbouring nodes exchange their magnitude values and, then, 'elect' the vanguard. To this end, we adopt the 'cluster-head' election strategy discussed in [Younis and Fahmy 2004] for a node i to be elected as vanguard. The strategy in [Younis and Fahmy 2004] meets the previously set of requirements and through some minor modifications in the original algorithm of [Younis and Fahmy 2004] we obtain a quick and energy-aware election method for use in our scheme.

At each node, the election process requires a number of iterations $K > 0$. In every step, nodes send and receive specific small-sized messages from neighbors. Before a node i starts the election process, it configures its probability of becoming a vanguard ξ_i, hereinafter referred to as **Election Probability** (EP), as a function of the sensed magnitude value of the phenomenon ϕ and to the current residual energy, i.e.,

$$\xi_i \sim \max\left(\xi_{\min}, \omega\frac{\mu_{\phi,i}(t)}{\mu_{\max}} + (1-\omega)\frac{\mathcal{E}_{res}(t)}{\mathcal{E}_{\max}}\right) \tag{21}$$

where $\mathcal{E}_{res}(t)$ is the current residual energy in the node and $\mathcal{E}_{\max}$ is a reference maximum energy (corresponding to a fully charged battery)[4]. The weighting factor $\omega \in [0,1]$ captures the relative importance of a node i to become a vanguard due to high residual energy or due to high magnitude value of the sensed phenomenon. On the one hand, we require that the node must have specific energy budget since, if elected as vanguard, it has to derive the optimal RDs by locally invoking the CPSO algorithm and, then, transmits the RDs to its members. On the other hand, we require that the elected vanguard should sense the highest magnitude value of a phenomenon compared to its neighboring nodes. Moreover, the EP value ξ_i of a node i is not allowed to fall below a certain threshold $\xi_{\min}$, e.g., 10^{-3} selected to be inversely proportional to $\mathcal{E}_{\max}$ as in [Younis and Fahmy 2004]. This restriction is essential for terminating the election process in $K = O(1)$ iterations, as show below.

A node i with a relatively high EP ξ_i starts the following process: it sends announcement messages of the form $\langle \xi_i, i\rangle$ to the $\mathcal{N}_i$ neighbours for being a vanguard. On the other hand, a node j with a low EP ξ_j delays the transmission of announcement messages and considers itself 'non-vanguard' if it has heard from $\langle \xi_i, i\rangle$ with $\xi_i > \xi_j$. Specifically, during iteration $k, 1 \le k \le K$, every node i decides to become a vanguard with EP ξ_i. Through the process, the node i can either be elected to become a vanguard according to its ξ_i or remain at the same status (i.e., non-vanguard) according to overheard announcement messages within its communication range. A node j selects its vanguard i to be the node with the highest residual energy and magnitude value; this is achieved by the comparison of the ξ_i and ξ_j values. Every node i then multiplies its EP ξ_i value with a factor of $\chi > 1$, and goes to the next step $k+1$ and so on, i.e., $\xi_i(k+1) = \min(\chi\xi_i(k), 1)$. If a node i decides to become a vanguard since its EP ξ_i value has reached 1, it sends, then, an announcement message 'vanguard i' to its neighbors in $\mathcal{N}_i$. A node $j \in \mathcal{N}_i$, then, considers itself 'non-vanguard' if it has heard from node i a 'vanguard i' message and terminates the election process.

Note that, as we reported above, this election process is completely distributed. A node can either decide to become a vanguard, since its weighted sum of the residual energy and the phenomenon magnitude value is the highest among its neighbors with respect to the EP of vanguard ξ_i, or be a member which awaits for RDs by its unique vanguard. In the election process in [Younis and Fahmy 2004] the weight factor ω in the EP ξ in (21) is $\omega = 0$ since it focuses exclusively on residual energy. In addition, the multiplication factor χ over the probability value between successive steps of the election process is set to 2.

LEMMA 6.2. *The election process requires $O(1)$ iterations.*

PROOF. Consider a probability multiplication factor $\chi > 1$ and that a node i starts with the minimum EP of being a vanguard, i.e., $\xi_i = \xi_{\min} > 0$. Since at each iteration step the node just multiplies its current EP ξ_i with χ then, in the worst case, that node will be either a vanguard or a member when the process stops at the first iteration

[4]The election process handles heterogeneous node batteries since every node has its own $\mathcal{E}_{\max}$value.

step K such that $\chi^{K-1}\xi_{\min} \geq 1$. That is, the maximum number of iteration steps are

$$K = \min\{k > 0 : \chi^{k-1}\xi_{\min} \geq 1\}.$$

Hence, the required number of iterations is $K = \lceil \log_{\chi} \frac{1}{\xi_{\min}} \rceil + 1$, which maps to $O(1)$ iterations. Now, if node i starts the election process with $\xi_i > \xi_{\min}$ given in (21) then $O(1)$ iterations are the maximum number of steps for the election process, for all ω values. □

Note that the number of iterations for each node does not depend on the number of neighboring nodes and is bounded by a constant. Indicatively, when $\xi_{min} = 10^{-3}$ and $\chi = e$ then a node needs at most eight iterations to elect or be elected as a vanguard. Moreover, for $\omega = 0.5$, i.e., we are equally interested in energy consumption and phenomena localization, for $\mu_{\phi,i} \to 1$ and for $\mathcal{E}_{res}(t) \to \mathcal{E}_{\max}$ then node i will terminate the election process earlier than nodes with lower residual energy, thus, allowing low energy nodes to join their vanguards.

LEMMA 6.3. *The message exchange complexity in the election process is $O(1)$ per node and $O(|\mathcal{N}|)$ in the network.*

PROOF. In the election process, a node which is about to become a vanguard generates at most $K = O(1)$ messages. On the other hand, a node which is about to become a member delays in sending messages and sends one message to just join its vanguard after considering itself as 'non-vanguard'. Obviously, the number of those messages (member messages) is strictly less than $|\mathcal{N}|$, since at least one node will decide to be a vanguard. Hence, the number of messages exchanged in the network is upper-bound by $K \times |\mathcal{N}|$, which is $O(|\mathcal{N}|)$. □

It is also worth noting that when a node j realizes that it is not the vanguard thus being assigned to the vanguard i after receiving the 'vanguard i' message it has to send a message to vanguard i with its current location coordinates. Such coordinates are gathered by the vanguard for executing the CPSO algorithm to proceed with the optimal RDs. The total number of messages exchanged during the election process is $O(1)$ per node (see Lemma 6.2 and 6.3), thus $O(|\mathcal{N}|)$ for the entire MSN.

6.3.3. Fault Tolerance. Right after the appointment of a node i as a vanguard, it locally invokes the CPSO algorithm based on the current location coordinates of its members and disseminates the RDs to them. Then, the members relocate and a new election process starts off. The election process is periodically triggered to elect a new vanguard, thus, balancing energy consumption among nodes along with moving swarms of nodes towards to the phenomena locations. The lifetime of a vanguard is relatively small since it only refers to the CPSO execution and the dissemination of RDs to the members. Since the idea of our model is to relocate the nodes around their vanguard i for better phenomena localization, then, with high probability, another node from the vanguard neighborhood $\mathcal{N}_i$ will become the new vanguard in the next election process; see Figure 6. Recall that the sensing discs of all members enclose the sensing disk of their vanguard after relocation as shown in Figure 1. Hence, there might be another neighboring node $j \in \mathcal{N}_i$ other than the vanguard i, which will be relatively closer, after the relocation, to the same phenomenon sensed also by the vanguard (refer to the domino-like process in Figure 7).

During the lifetime of a vanguard there might be unexpected failures of nodes, especially failures of the vanguards. This may cause parts of the MSN to be unable for phenomena localization. Nonetheless, after this relatively short lifetime period, a new election process takes place with the current available (operational) nodes, which might be possibly less in number due to certain failures. This evidently would decrease

the quality of phenomena localization and identification process, at least for a certain period of time (e.g., until nodes replacement or recharging). Specifically, in the event of a vanguard failure, e.g., the vanguard runs out of energy, the vanguard i can either notify its members in $\mathcal{N}_i$ prior to shutdown (graceful removal) or not. In order to promote the uninterruptible operation of the proposed scheme, despite possible failures of the appointed vanguards (e.g., energy exhaustion), we propose two strategies: the 'soft' and 'hard' vanguard replacement.

We introduce a fault tolerance policy, during the lifetime of a vanguard i, such that specific vanguard deputy nodes from $\mathcal{N}_i$ are determined to manage unexpected failures of the current vanguard. Such deputies will be undertaking the vanguard responsibilities following a vanguard failure event. This, obviously, mitigates the single point of failure problem at a vanguard i, since a member node $j \in \mathcal{N}_i$ (which is appointed as a deputy of vanguard i) can take the leadership. A criterion of a member from $\mathcal{N}_i$ to be vanguard deputy is that node $j \in \mathcal{N}_i$ has the second largest EP ξ_j compared to that of vanguard i. This information is locally known to node j, since, based on the election process, each node knows its own EP ξ_j value and that of its vanguard ξ_i. The trade-off in this case is that the vanguard deputy j is not the 'best' node in terms of residual energy and phenomena magnitude. Instead, it is the second best among the remaining members which is, therefore, operational. Obviously, we can further extend this policy to the top-r members whose EP values ξ_j, $j = 1, \ldots, r$ refer to the $|\mathcal{N}_i| - r + 1$-order statistics, i.e., their EP values are the r highest values among all $|\mathcal{N}_i|$ EP values. We now proceed with the two policies of replacing the current vanguard with the vanguard deputies.

Hard vanguard replacement: In this policy, none of the members are notified upon a failure event of the vanguard i. Consider the top-r node j, which knows right after the election process that it is a 'non-vanguard' and assigned to the vanguard i. Then, node j waits for a RD from its vanguard i a certain time horizon. This time horizon is proportional to the rank r of each node's EP value plus the expected time required for a vanguard to execute the CPSO algorithm as shown in Figure 3. If this horizon expires then it automatically considers itself as the vanguard deputy assuming that vanguard i is no longer operational and immediately sends to its neighbors a 'vanguard deputy j' message. Consequently, node j requests from the remaining nodes their location coordinates to locally execute the CPSO and the process evolves as described above.

Soft vanguard replacement: This policy copes with a graceful removal upon a failure event of the vanguard i. Specifically, the vanguard i before ceasing to operate, e.g., by identifying that from its residual energy levels, notifies its members about its unavailability together with a 'vanguard deputy j' message. That is, vanguard i keeps the top-r list of members with respect to their EP values and assigns, before its retirement, as a vanguard deputy the top-r member j. Then, the node j after receiving this message from vanguard i, takes over the vanguard leadership. Each node $k \in \mathcal{N}_i \setminus \{j\}$ upon reception of the 'vanguard deputy j' message sends its location coordinates to the newly appointed vanguard deputy j. Then, the whole process follows as discussed above. Note that, if node j is not responsive then vanguard i proceeds with another attempt of notifying the member with the third highest EP value. This notification process continues until all r members are notified, which is the worst case scenario.

6.4. Nodes Reorganization

Two strategies can be adopted for MSN reorganization. One strategy could involve the automatic return of nodes to their initial positions. This is strongly dependent upon the nature of the observed phenomenon and the energy budget of nodes. Should the phenomenon demonstrate a localized nature, a nodal relocation scheme to original

positions would unnecessarily reduce the energy budget and increase the reaction in a new incident occurrence. On the other hand a 'spatial freeze' approach after the event occurrence would not be a sound approach if new incidents appear uniformly in the terrain F. The 'spatial freeze' approach (until a new detection) is adopted in this paper. Moreover, a node does not move the entire distance from its starting position to the POI of the phenomenon. The domino-like fashion of the algorithm requires that nodes make small movements (at most the ϵ value in distance) since they exchange location and magnitude information among temporally elected vanguards.

It is worth noting that the election process does not deal with the uncertainty and measurement errors of the magnitude values. As mentioned above, nodes estimate locally the received magnitude value in order to reason about the occurrence of a phenomenon in a probabilistic way. However, the inherent uncertainty on the measurements might lead to misleading EP values ξ_i. The proposed method has to be enhanced with techniques that take into consideration a degree of uncertainty on the measurements of the magnitude values. Furthermore, vanguards determine the RDs based on their measurements, while nodes receiving such RDs might lead to a neighborhood split, which could be badly initiated by measurements error. It is of high importance to deal with such uncertainty, which affects the robustness of the proposed scheme. The adoption of an approximate reasoning algorithm in the election process and in the exploitation of the RDs by a node is a future research item.

7. PROCESS COMPUTATIONAL COMPLEXITY & ENERGY COST

In this section, we report on the total energy cost (communication, computation, and mobility energy consumption) of a node being involved in the distributed phenomena localization process and of the MSN as a whole. We provide the computational complexity for all the sub-processes and, also, demonstrate the actual energy cost induced per node based on the experimental results as they will be shown in Section 8.4.

7.1. Computational Complexity

Let us focus on a single node i, which is involved in distributed phenomena localization process. Recall that the whole sequence of the sub-processes as discussed in Section 6.1 for node i, which, hereinafter, is referred to as **process era**, is as follows: (i) the node i starts an election process; (ii) if node i is elected as a vanguard, it performs the CPSO coverage algorithm; (iii) if node i is a vanguard, it sends relocation directives to its neighbours; (iv) if node i is a 'non-vanguard' member, it relocates to the specified location dictated by the RD message. Based on the above sub-processes, we provide the corresponding complexity for both types of nodes: vanguard and non-vanguard. The election process in Section 6.3 for a node i requires $O(1)$ iterations to end (computational time) and $O(1)$ messages to be sent (communication cost). If node i is not a vanguard then it sends its location coordinates to its vanguard, thus, $O(1)$ communication cost. If node i is a vanguard then the optimal coverage process (i.e., CPSO algorithm over $F(\ell_i, \epsilon)$) requires $O(nm\lceil\epsilon/a\rceil^2 T)$ time with $n = |\mathcal{N}_i|$. The node i, which is non-vanguard, upon reception of a RD, it requires $O(\epsilon)$ walking distance (mobility cost) for being relocated to the position dictated by the RD. This is the worst case scenario at which a member node i moves all the diameter of the connectivity disc of its vanguard i with radius ϵ as shown in Figure 1. However, as discussed in Section 5 the relocation directives are not randomly disseminated to the members. Instead, the vanguard takes into consideration to relocate a members with the minimum possible traveled distance, thus, decreasing the incurred energy mobility cost. The total complexity of a process era for the MSN with $|\mathcal{N}|$ nodes and $|\mathcal{V}| < |\mathcal{N}|$ vanguards is: $O(|\mathcal{N}|)$ messages and iterations for the election process, $O(\frac{|\mathcal{N}|}{|\mathcal{V}|} m\lceil\epsilon/a\rceil^2 T)$ for the coverage CPSO algo-

rithm running on the vanguards, and $O(\epsilon|\mathcal{N}|)$ walking distance of the non-vanguard nodes. The process era is repeated until all phenomena are identified and localized by the MSN. Table III summarizes the asymptotic complexities of the entire process for both type of nodes (vanguards and non-vanguards/members) per sub-process.

Table III. The asymptotic complexities for each sub-process per node; n refers to the neighborhood size of a vanguard and '-' means 'not applicable'.

Sub-process	Node type	Communication	Computation	Traveled distance
Election	vanguard	$O(1)$	$O(1)$	-
	member	$O(1)$	$O(1)$	-
Coverage	vanguard	-	$O(nm\lceil\epsilon/a\rceil^2 T)$	-
	member	-	-	-
Relocation	vanguard	$O(1)$	-	-
	member	$O(1)$	-	$O(\epsilon)$

7.2. Communication, Computation, & Mobility Energy Consumption

Mobile nodes must accomplish their assigned sensing tasks by using the limited energy resources carried by them. The energy refers to a number of operations: mobility, wireless communication, sensing the environment, and computation. Among them, motion and wireless communications are the two major concerns of the node (robot) energy, apart from the processing power. The authors in [Labella et al. 2006] state that the energy required for node movements is generally bigger than for communications. While the mobility cost grows linearly with the traveled distance, the energy consumption of wireless communication grows at least quadratically with the distance of two communicating nodes.

In our study, the energy model reflects three facets: energy for communication, energy for mobility, and energy for computation/processing. Specifically, the energy model for the communication takes into account the distance between two nodes, and other factors like interferences, multi-path fading, and other noises in the transmission medium. The energy consumed to transmit x bits of data over distance d measured in meter is

$$\mathcal{E}_T = x(d^y e_{TX} + e_{CT}), \tag{22}$$

where e_{TX} is the energy required by the power amplifier of transceiver to transmit one bit data over the distance of one meter, and e_{CT} is the energy consumed in the electronic circuits of the transceiver to transmit or receive one bit measured in the unit of Joule/bit. Based on the transceiver sensitivity, the value of e_{TX} ranges from pico- to nano-Joule per bit per meter^y, with $y \in [2, 6]$ be the loss exponent of the transmission medium expressing different types of environment in which the wireless communication takes place. In addition, the energy consumption for receiving x bit of data is

$$\mathcal{E}_R = x e_{RX}, \tag{23}$$

which is independent of the distance between communicating nodes and e_{RX} is the parameter reflecting energy to receive a bit in Joule/bit.

We adopt the energy mobility model for mobile robots from [Yongguo et al. 2005] and [Chiping & McKinley 2006]. The mobility energy depends on the mass of the node, the friction to the surface (air or ground), gravity and acceleration, and the distance traveled. For simplicity, we focus on the energy model that is proportional to the traveled distance as in [Chiping & McKinley 2006], which is reasonable for wheeled robots, and is defined as

$$\mathcal{E}_M = \nu d, \tag{24}$$

where the movement parameter ν measured in Joule/m, is constant based on the aforementioned factors, and d is the traveled distance in meter by the node. Finally, each node consumes processing power for executing the CPSO algorithm, when this node becomes a vanguard, and for computing its EP value based on the captured magnitude values. We notate with $\mathcal{E}_P$ the is the energy cost in Joule per CPU instructions corresponding to an executable algorithm.

During a process era, the total energy cost $\mathcal{E}_i$ in Joules is the energy cost incurred on node i by transmitting and receiving election messages, executing the CPSO algorithm and transmitting relocation directives if node i is vanguard, receiving a relocation directive and relocating/traveling to the optimal location if node i is a member, i.e.,

$$\mathcal{E}_i = \mathcal{E}_T + \mathcal{E}_R + I_i\mathcal{E}_P + (1 - I_i)\mathcal{E}_M + \mathcal{E}_0, \tag{25}$$

where $\mathcal{E}_0$ is the energy cost for node i transiting from idle to standby operational modes [He et al. 2004] and I_i is an indicator function with $I_i = 1$ if node i is elected to be a vanguard at the process era, otherwise $I_i = 0$. In Section 8.4 we extensively report on the total cumulative energy cost per node when involved in the proposed process over certain experimental scenarios (Section 8).

7.3. Model Improvement

In this section, we discuss factors that can improve the complexity of the proposed coverage sub-process focusing on decreasing the consumed computation energy per node. As discussed in Section 7.1, we could further decrease the computational complexity of the coverage sub-process, which primarily depends on the number of iterations T and the discretization factor a. In Section 5.3, we provide a discussion on the impact of factor a in the time complexity of the CPSO algorithm depicting that a small increase in a (i.e., a quite coarse discretization) renders a quadratic decrease in the time complexity without significant changes in the estimation of the coverage probability. Hence, through fine tuning of the parameter a we can achieve acceptable levels of estimation of the $J(\mathbf{p})$ fitness value. With respect to the number of iterations T, in Section 5.2, we provide an analysis on the required number of iterations T^*_λ, which depends on the coverage radius ζ given a fixed connectivity radius ϵ for a node. Obviously, when $\zeta \to \epsilon$ then T^*_λ decreases too, as derived by Eq(17). This is due to the fact that we are interested in covering a disc area with radius $\zeta \to \epsilon$ with almost equal area with the connectivity disc area of the vanguard. This denotes that the coverage of that ROI might not be possible, especially in the case where the number of the neighboring nodes n is low. Consider the case to cover a relatively huge area with a relatively low number of nodes. Evidently, in that case, the degree of coverage $J(\mathbf{p})$ will be low. Hence, a ζ value should be relatively low with respect to ϵ value to guarantee an acceptable level of the degree of coverage. Through tuning the ratio $\frac{\zeta}{\epsilon}$, we can 'control' the required number of iterations for the CPSO algorithm to provide us a relatively high degree of coverage given a specific number of neighbors n. It is worth noting that a small increase in ζ results in a quadratic decrease of the number of iterations T^*, thus, we can tolerate a 'near' optimal solution with the benefit of decreasing the time complexity of the CPSO in terms of iterations. Finally, since the coverage sub-process relies on the CPSO algorithm, one can adopt certain computational improvements on its corresponding PSO algorithm (especially in terms of iterations and number of velocity updates) yielding almost same optimal results as proposed in [Wu et al. 2011], [Li et al. 2013], [Deb & Padhye 2010], [Pan & Liu 2011], and the 'competitive' and 'social learning' PSO variants in [Cheng & Jin, 2015a] and [Cheng & Jin, 2015b], respectively.

8. PERFORMANCE EVALUATION

We study the performance of our proposed approach in terms of realistic scenarios, in which certain phenomena are randomly occur, with respect to the (i) degree of coverage of the phenomena, (ii) the traveled distance of the nodes to localize the phenomena, (iii) the closest distance of nodes to the phenomena, and (iv) the expected number of vanguards required to lead their members to localize the random phenomena. Moreover, for each of the scenarios, we study the total excepted cumulative energy consumption per node in terms of (i) mobility energy cost, (ii) communication cost, and (iii) computation and sensing cost.

8.1. Simulation setup

We present simulation results of the proposed process. All distances are measured in meters. The are under investigation F is a 500m $\times 500$m terrain and the number of nodes $|\mathcal{N}| \in \{50, 100, 125, 150\}$. Initially, nodes spread randomly in F. We set sensing radius $\delta = 3$m, range detection error $\delta_e = 1$m and $(\alpha_1, \alpha_2, \beta_1, \beta_2) = (1, 0, 0.5, 0.5)$ for the probabilistic detection model. The communication range is $\epsilon = 50$m and the coverage ROI radius $\zeta \in [0.2\epsilon, 0.7\epsilon]$. The rationale behind a value of $\epsilon = 50$m is based on the current wireless communication technology, where for a path loss exponent between 2 and 4 (in our case, we set the path loss to 4; see Section 8.4), we obtain a communication range for IEEE 802.11, 25–600 m, for Bluetooth, 10–100 m, Zig-Bee, 10–75 m, HomeRF, 50 m, etc. Moreover, we require a relatively high communication range in order for a vanguard node to gather more members for improving the ROI coverage around it, whose radius ζ is bounded in (δ, ϵ). But, on the other hand, we also require this communication range to be relatively small to avoid high communication overhead and a redundant number of nodes covering the ROI. In addition, the value of ϵ also depends on the sensing radius δ. As we discussed in Section 3.2 and according to [Zhang et al. 2005] by setting $\epsilon \geq 2\delta$; Theorem 1 in [Zhang et al. 2005] (or $\epsilon \geq \sqrt{3}\delta$; Theorem 4 in [Zhang et al. 2005]) it is both necessary and sufficient to ensure that coverage implies connectivity; this yields a minimum value of $\epsilon_{\min} = 6$ m with respect to a sensing radius $\delta = 3$ m. Given that we desire a communication range that satisfies the above mentioned characteristics. In addition, since we deal with the POI phenomena localization in the RMCAA domain, we require thorough coverage of the ROI and, thus, we set the sensing radius $\delta = 3$m, which is also widely used for wild fire front identification in [Sekkas et al. 2010] and SCIER System[5]. We set $m_1 = 60, m_2 = 64$ for the adopted radiation flux in Eq(5) (in kW/m^2) as modeled in [Manolakos et al. 2008]. The number of particles for the CPSO algorithm is $m = 20$ as adopted in [Wang et al. 2007, Azlina et al. 2010]; see also Section 5.1. In addition, we adopt $(w_{max}, w_{min}) = (0.9, 0.4)$ for the CPSO as proposed in [Shi and Eberhart 1999], maximum number of generations $T = 100$ (see the analysis in Section 5.2) and $(c_1, c_2, c_3) = (1.4, 1.4, 1.2)$. Moreover, the f value in the WRP depends on the γ parameter. We consider that a node j follows the RD from its vanguard node i with a weight of $f = 0.95$ adopted by Eq(19) if the percentage difference of the magnitude values between node j and node i are over 100%. In that case, the f value is 0.95 with $\gamma = 0.57$. The default simulation parameters are grouped based on the corresponding model and summarized in Table IV.

8.2. Performance Metrics

We define the metrics for evaluating the performance of the proposed process.

[5]SCIER - Sensor & Computing Infrastructure for Environmental Risks; EU IST-5-035164 (FP6); http://www.scier.eu/

Table IV. Simulation parameters.

Parameter	Description	Default Value/Range
	Sensor field	
F	Area of the sensor field	500m $\times$ 500m
$\|\mathcal{N}\|$	Number of nodes in F	$\{50, 100, 125, 150\}$
γ	Relocation directive weight	0.57
$\|\Psi\|$	Number of simultaneous phenomena (Scenario III)	$\{1, 2, 4, 8, 12\}$
σ	Mean of exponential duration of sequential phenomena (Scenario IV)	0.08
α	Probability of phenomenon occurrence (Scenario V)	$\{0.02, 0.06, 0.10, 0.14\}$
	Sensing model	
δ	Sensing range	3m
δ_e	Range detection error	1m
$(\alpha_1, \beta_1, \beta_2)$	Probabilistic parameters	$(1, 0.5, 0.5)$
α_2	Disturbing effect	0
	Communication model	
ϵ	Communication range	50m
	Phenomenon model	
(m_1, m_2)	Radiation flux in kW/m^2	$(60, 64)$ [Manolakos et al. 2008]
	CPSO algorithm	
m	Population size	20 (Section 5.1)
(w_{max}, w_{min})	max. and min. inertia weights	$(0.9, 0.4)$
(c_1, c_2, c_3)	acceleration coefficients	$(1.4, 1.4, 1.2)$
T	Maximum number of iterations	100
a	Discretization factor	0.5m
ζ	Coverage radius	$[0.2\epsilon, 0.7\epsilon]$ (default: 0.35ϵ)
ρ	Convergence threshold for minimum required iterations	10^{-5}
	Election process	
$\xi_{\min}$	Minimum election probability threshold	10^{-3}
χ	Multiplication election probability factor	2
ω	residual energy and magnitude weighting factor	0.5

— The **traveled distance** $\mathcal{D}(t)$ is defined as the mean distance that all nodes traveled up to t. $\mathcal{D}$ should assume a rather constant value once all phenomena have been localized, thus, there is not need for nodes to move further. Let us define the trajectory $\mathcal{T}_i(t)$ of node i from its initial location $\ell_i(0)$ up to t as the tuple: $\mathcal{T}_i(t) = \langle \ell_i(0), \ldots, \ell_i(t) \rangle$. The traveled distance up to t is $\mathcal{D}_i(t) = \sum_{\tau=1}^{t} \parallel \ell_i(\tau) - \ell_i(\tau - 1) \parallel$. Hence,

$$\mathcal{D}(t) = \frac{1}{|\mathcal{N}|} \sum_{i \in \mathcal{N}} \mathcal{D}_i(t).$$

— We define as **degree of phenomena coverage** $\mathcal{C}(t)$ the percentage of the covered $F(\ell_\phi, \zeta)$ areas corresponding to observed phenomena $\phi \in \Psi(t), \Psi(t) \neq \emptyset$, (occurred at ℓ_ϕ POI) by nodes. Let $\mathcal{F}(F(\ell_\phi, \zeta)) \subseteq F(\ell_\phi, \zeta)$ be the subset of q points, $q \in F(\ell_\phi, \zeta)$, which are covered by some node $i \in \mathcal{N}$ with detection probability $\theta_q(\ell_i) > 0$, i.e., $\mathcal{F}(F(\ell_\phi, \zeta)) = \{q \in F(\ell_\phi, \zeta) : \exists i \in \mathcal{N} \text{ with } \theta_q(\ell_i) > 0)\}$. Hence,

$$\mathcal{C}(t) = \frac{1}{|\Psi(t)|} \sum_{\phi \in \Psi(t)} \frac{|\mathcal{F}(F(\ell_\phi, \zeta))|}{|F(\ell_\phi, \zeta)\}|}.$$

The cardinality of the $\mathcal{F}(F(\ell_\phi, \zeta))$ set is approximated through the Monte Carlo method [Averil and Kelton 2000] with grid granularity of 0.1m (distance between grid points on each coordinate). $\mathcal{C}(t)$ should assume value close to unity denoting that all vanguards and their neighbours determine accurately the POIs.

— We define as **phenomena closest distance** $\mathcal{A}(t)$ the mean distance of the closest vanguards to the detected $\phi \in \Psi(t)$, i.e., to the corresponding ℓ_ϕ POI. Let $i_\phi(t) \in \mathcal{V}(t)$

be the closest vanguard to ϕ, i.e., $i_\phi(t) = \arg\min_{j \in \mathcal{V}(t)} \parallel \ell_j - \ell_\phi \parallel$. Then, the set of the vanguards that are closest to $\phi \in \Psi(t)$ is $\mathcal{U}_{\Psi(t)}(t) = \{i_\phi(t) : \phi \in \Psi(t)\}$ and, then,

$$\mathcal{A}(t) = \frac{1}{|\mathcal{U}_{\Psi(t)}(t)|} \sum_{i_\phi(t) \in \mathcal{U}_{\Psi(t)}(t)} \parallel \ell_{i_\phi(t)} - \ell_\phi \parallel .$$

The $\mathcal{A}(t)$ metric indicates the mean (remaining) distance of the closest vanguards to the phenomena. $\mathcal{A}(t)$ should be at most δ ($\mathcal{A}(t) \leq \delta$) indicating that vanguards are located at the phenomena sites.

— We examine the capability of the algorithm to have vanguards merged their neighbourhoods in order to monitor the same phenomenon. This can be investigated by the **current number of vanguards** $|\mathcal{V}(t)|$. Specifically, $|\mathcal{V}(t)|$ should assume a value equal to the number of phenomena $|\Psi(t)|$. This indicates that each phenomenon is covered exactly by one vanguard.

8.3. Performance Scenarios

We consider the following realistic scenarios in order to evaluate the performance of our algorithm: **Scenario I** – a unique phenomenon appears within the area F; **Scenario II** – a unique phenomenon appears outside the area F; **Scenario III** – a number of phenomena occur simultaneously. **Scenario IV** – a given number of phenomena occur progressively at given time instances each one with an exponential duration lifetime (until it perishes); **Scenario V** – a number of phenomena appear progressively at random time instances with certain probability of occurrence. We performed 50 simulation runs for each scenario.

Remark 8.1. The default values for the simulation parameters used in the following experiments are shown in Table IV unless otherwise stated.

Scenario I: unique phenomenon occurrence inside ROI. We examine the behaviour of the algorithm when a phenomenon ϕ occurs at a random location $\ell_\phi \in F$. For demonstration purposes, Figure 10 illustrates the trajectories of the vanguards towards the unique phenomenon ϕ. Vanguards are coming from different directions approaching the phenomenon ϕ. Nodes exchange vanguard relationships in a domino-like fashion until starting to merging their neighbourhoods once they come close to ϕ's location. We can observe that phenomenon ϕ, finally, is localized by one vanguard, thus, illustrating the merging capability of the algorithm.

Figure 11 shows the degree of phenomenon coverage $\mathcal{C}(t)$ and phenomenon closest distance $\mathcal{A}(t)$ for different number of nodes $|\mathcal{N}|$. Note that the rest parameter values are set to default as shown in Table IV. The phenomenon is covered 100% within $t = [5, 10]$ once there are $|\mathcal{N}| > 50$. Moreover, we can observe from Figure 11 (right) that for $t > 10$ and $|\mathcal{N}| > 50$ the phenomenon is covered by at least one vanguard, and, especially, the vanguard is accurately localized on the phenomenon location ($\mathcal{A} < \delta$). For $|\mathcal{N}| = 50$ the algorithm achieves phenomenon coverage at $t > 15$.

Figure 12 shows the traveled distance $\mathcal{D}(t)$ and current number of vanguards $|\mathcal{V}(t)|$ for different number of nodes $|\mathcal{N}|$, while the rest parameter values are set to default. One can observe that the mean traveled distance of nodes converges, once the phenomenon is located. We can also observe that through the domino-like operation of the algorithm the mean traveled distance is relatively small ($\mathcal{D} \in [9, 14]$m). That is because each node contributes temporarily to the phenomenon detection by covering an area. After that, another candidate vanguard gathers neighbours and so on until reaching the phenomenon ϕ. Evidently, a large number of nodes reduces the mean traveled distance for locating the phenomenon. Figure 12 (right) shows that the number of vanguards decreases with time, thus, indicating the merging of neighbourhoods. It is

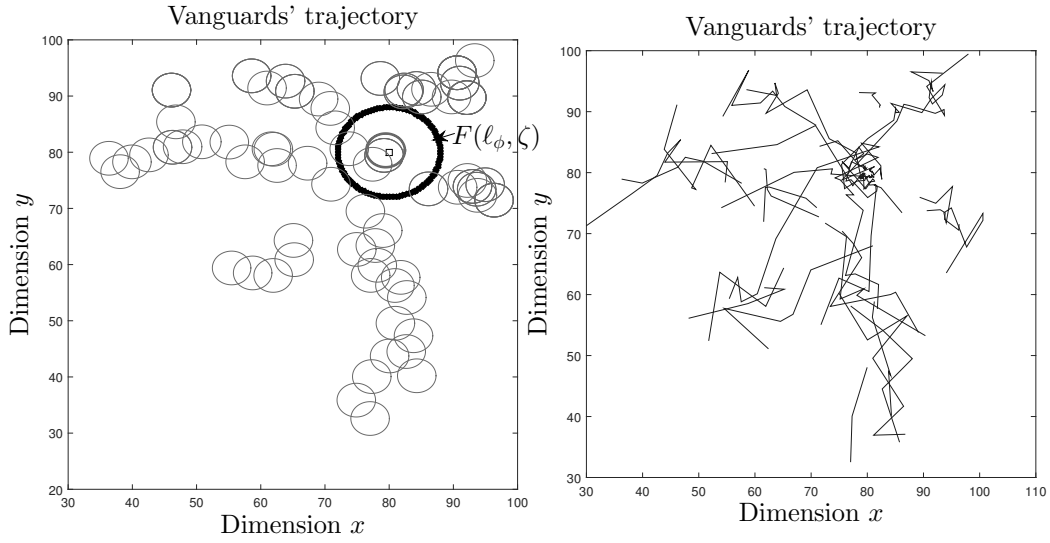

Fig. 10. Scenario I: The vanguards' trajectories toward a phenomenon ϕ inside the ROI.

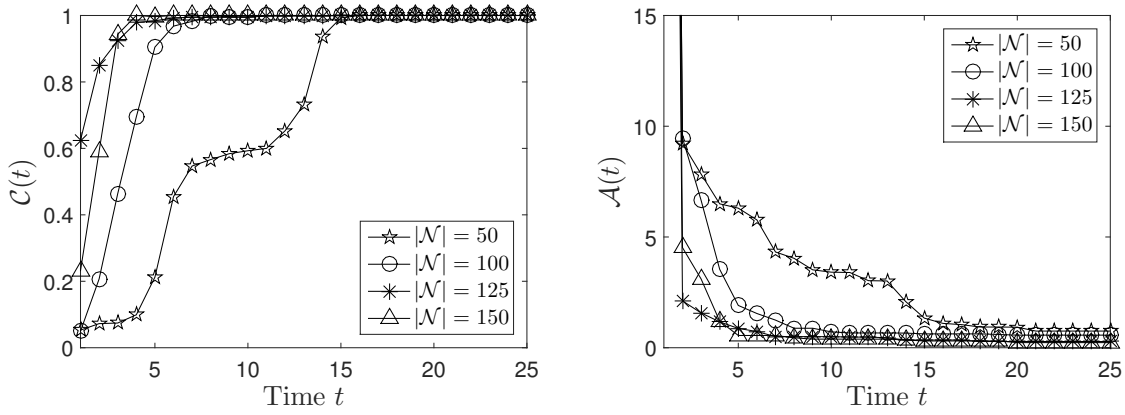

Fig. 11. Scenario I: (left) the degree of phenomenon coverage $\mathcal{C}(t)$ against time t, (right) the phenomenon closest distance $\mathcal{A}(t)$ against time t for different number of nodes $|\mathcal{N}| \in \{50, 100, 125, 150\}$; the rest parameter values are set to default as shown in Table IV.

worth mentioning that at the beginning of the monitoring phase, a certain number of first responders detect the phenomenon $|\mathcal{V}(0)| \simeq 8$. As time passes, the corresponding vanguards come close to ℓ_ϕ and, thus, merge their neighbourhoods. We obtain a mean value of $|\mathcal{V}| = 1.7$ when ϕ is accurately located. It is worth noting that the number of nodes $|\mathcal{N}|$ does not play significant role in the merging capability of the algorithm.

We also examine the behaviour of the algorithm for certain values of coverage radius ζ. Figure 13 shows the traveled distance $\mathcal{D}(t)$ (on the left) and the current number of vanguards $|\mathcal{V}(t)|$ (on the right) vs. time for different ζ values with respect to sensing radius δ and number of nodes $|\mathcal{N}| = 100$. The rest parameter values are set to default. A high ζ value indicates a broader sensing region of a vanguard. This results to

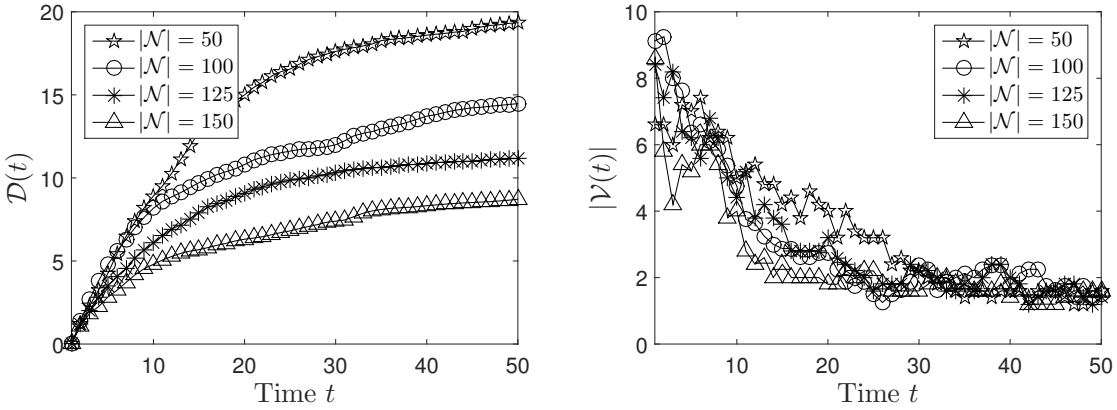

Fig. 12. Scenario I: (left) the traveled distance $\mathcal{D}(t)$ against time t; (right) the number of vanguards $|\mathcal{V}(t)|$ against time t for different number of nodes $|\mathcal{N}| \in \{50, 100, 125, 150\}$; the rest parameter values are set to default as shown in Table IV.

an enhanced environmental awareness of the broader area and, thus, rapid (spatial) convergence to the phenomenon location. However, the ζ value cannot be arbitrary high as discussed in Section 6.2. On the other hand, a small ζ value with respect to δ requires more vanguards to detect the same area compared to high ζ. As a result, the phenomenon localization process delays, since a high number of vanguards and neighbours, which approach phenomenon ϕ's location, exchange continuously vanguard leaderships. More interestingly, we observe that for $\zeta = 2.66\delta$ (highest value) $|\mathcal{V}| = 1$, $|\mathcal{V}| = 3$ for $\zeta = 1.66\delta$, and $|\mathcal{V}| = 7$ for $\zeta = 1.33\delta$ (Figure 13 (right)).

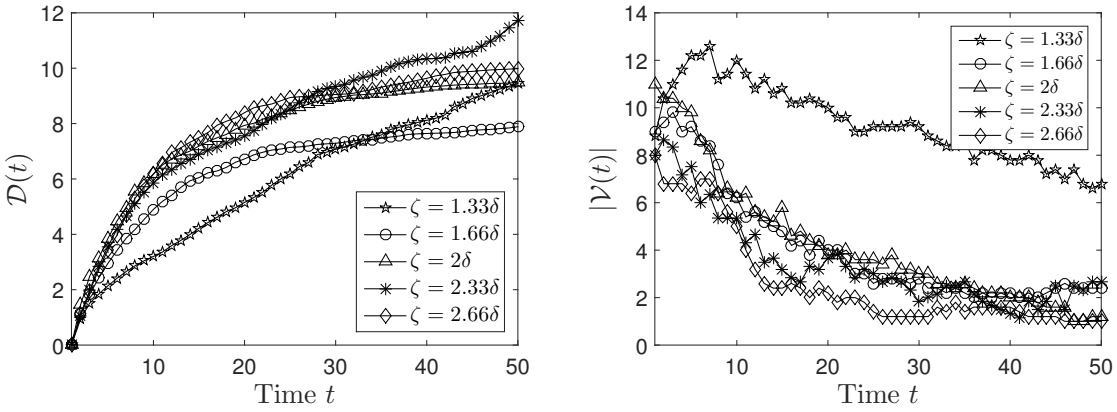

Fig. 13. Scenario I: (left) the traveled distance $\mathcal{D}(t)$ against time t; (right) the number of vanguards $|\mathcal{V}(t)|$ against time t for different coverage radius ζ values with number of nodes $|\mathcal{N}| = 100$; rest parameter values are set to default.

Scenario II: unique phenomenon occurrence outside ROI. Consider now that ϕ occurs outside F but it is detectable (with very low magnitude). For illustration reasons, Figure 14 shows the vanguards' trajectories towards to a phenomenon ϕ outside the ROI. We can observe the trajectory 'line' of the vanguards moving towards the phenomenon ϕ through certain split and merge of the 'swarms'. The phenomenon ϕ occurs

at a distance of 60m from the borders of F. In this case we assume that nodes can move outside F once they have detected some phenomena. Figures 15 and 16 show all the performance metrics ($\mathcal{C}(t), \mathcal{A}(t), \mathcal{D}(t), |\mathcal{V}(t)|$) for different number of nodes $|\mathcal{N}|$; the rest parameter values are set to default. The coverage of the phenomenon ϕ's POI starts at time instance $t = 12$ and ϕ is accurately localized at $t = 16, 22, 27$ for $|\mathcal{N}| = 150, 100, 50$, respectively. For $|\mathcal{N}| = 150$, we obtain 35% and 55% less traveled distance compared to $|\mathcal{N}| = 100$ and $|\mathcal{N}| = 50$, respectively. Moreover, the phenomenon is accurately localized independently of the number of nodes $|\mathcal{N}|$. One can observe that $\mathcal{A} \ll \delta$ at the end of the process. We can also observe the split and merging capability of the algorithm by examining the $|\mathcal{V}|$ metric as shown in Figure 14. At the beginning of the process, only some nodes close to the borders of F (first responders) detect the phenomenon. Then, as they gather neighbours for better investigating the area, more nodes become vanguards. Finally, as vanguards approach the phenomenon ϕ, they merge their neighbourhoods.

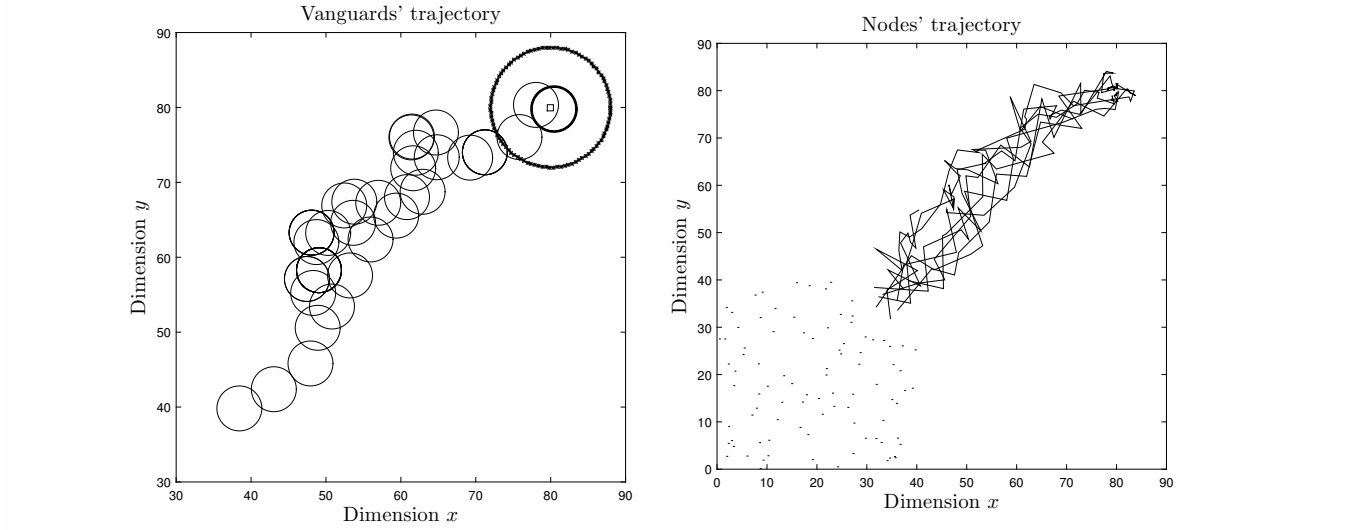

Fig. 14. Scenario II: The vanguards' trajectories toward the phenomenon ϕ outside the ROI.

Scenario III: simultaneous, multiple phenomena occurrences. We evaluate the robustness of the algorithm when dealing with multiple simultaneous phenomena. Figure 17 and 18 show the performance metrics for different number of simultaneous phenomena $|\Psi(t)|$ having $|\mathcal{N}| = 100$ with the rest parameter values set to default. We observe that the algorithm can accurately localize all phenomena, as depicted by the $\mathcal{A}$ metric, where the phenomena closest distance is less the that sensing radius, i.e., $\mathcal{A} < \delta$, for $|\Psi(t)| \in \{2, 4, 8, 12\}$. Moreover, the algorithm is robust with respect to the $\mathcal{C}$ metric. Especially, when $|\Psi(t)| = 12$, we obtain a degree of coverage $\mathcal{C} = 0.81$ and, for $|\Psi(t)| \in \{2, 4, 8\}$ the degree of coverage $\mathcal{C} = 1$. In addition, the traveled distance $|\mathcal{D}(t)|$ converges for time instance $t > 25$ for all $|\Psi(t)|$ values. Note that, the higher the number of simultaneous phenomena is, the lower the traveled distance of the nodes gets, once the phenomena occur within the F area. Finally, the number of vanguards

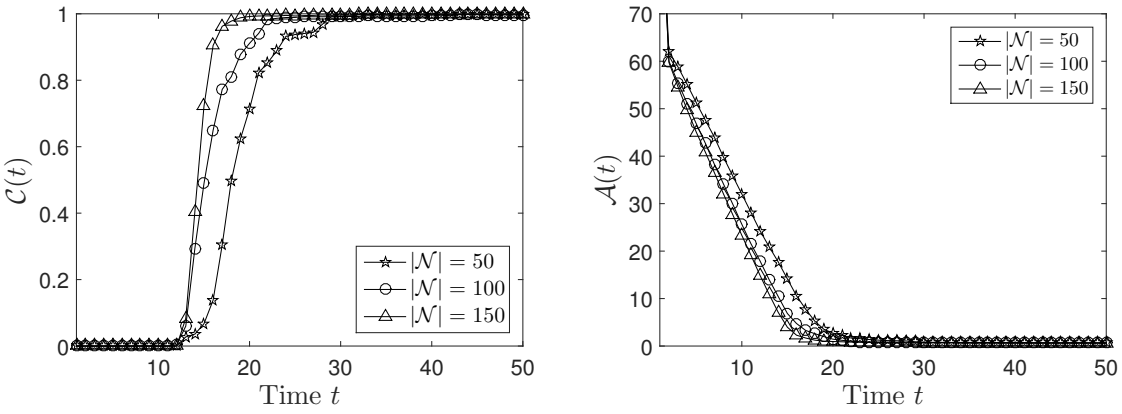

Fig. 15. Scenario II: (left) the degree of phenomenon coverage $\mathcal{C}(t)$ against time t, (right) the phenomenon closest distance $\mathcal{A}(t)$ against time t for different number of nodes $|\mathcal{N}| \in \{50, 100, 125, 150\}$; the rest parameter values are set to default as shown in Table IV.

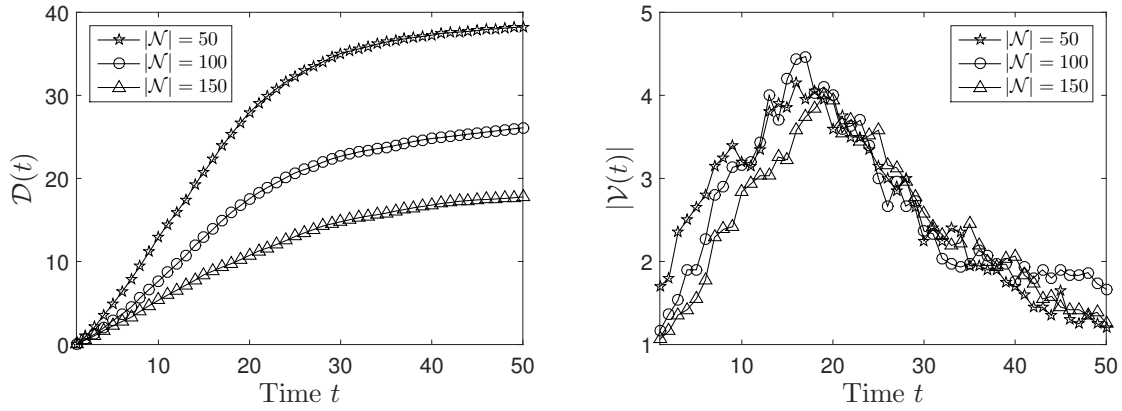

Fig. 16. Scenario II: (left) the traveled distance $\mathcal{D}(t)$ against time t; (right) the number of vanguards $|\mathcal{V}(t)|$ against time t for different number of nodes $|\mathcal{N}| \in \{50, 100, 125, 150\}$; the rest parameter values are set to default as shown in Table IV.

$|\mathcal{V}(t)|$ tends to the $|\Psi(t)|$ value, thus, depicting the capability of the algorithm to localize phenomena with the minimum number of vanguards.

Scenario IV: sequential, multiple phenomena occurrences. We experiment with the capability of the algorithm to adjust the RDs of vanguards to sequential occurrences of phenomena. In addition, such phenomena perish after an exponential duration. Imagine the case that a phenomenon can no longer be observed. Hence, we experiment with the self-organized nature of the proposed algorithm. Initially, a phenomenon ϕ occurs at a random point on the terrain. At instances $t_1 = 10, t_2 = 20$, and $t_3 = 30$, the phenomena ϕ_1, ϕ_2, and ϕ_3 occur (at random points), which last $T_1, T_2, T_3 \sim Exp(\sigma), \sigma = 0.08$, respectively. Figures 19 and 20 show all performance metrics for this scenario with $|\mathcal{N}| = 100$ number of nodes and all other parameter values are set to default. One can observe that $\mathcal{C}(t)$ tends to unity after each unexpected phenomenon occurrence denoting that the nodes are self-organized in order to cover each new phenomenon. Moreover, the nodes accurately localize each new phenomenon

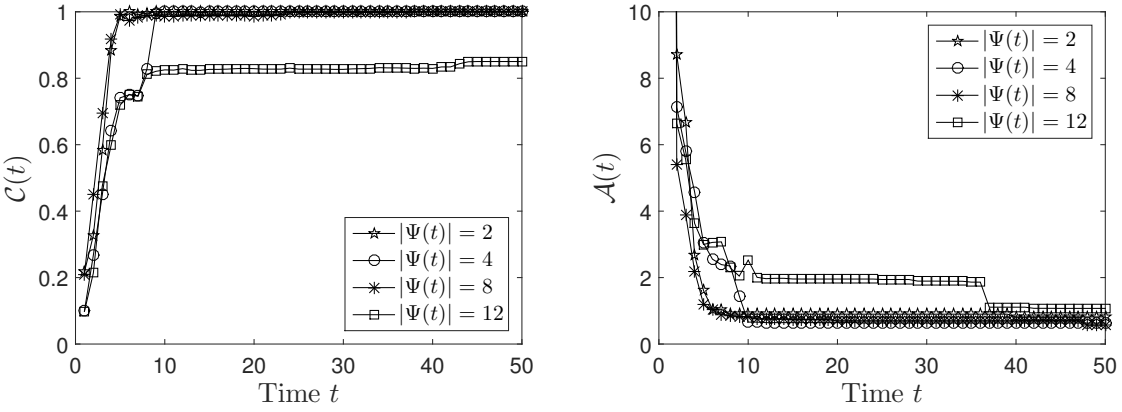

Fig. 17. Scenario III: (left) the degree of phenomenon coverage $\mathcal{C}(t)$ against time t, (right) the phenomenon closest distance $\mathcal{A}(t)$ against time t for different number of phenomena $|\Psi| \in \{2, 4, 8, 12\}$ and number of nodes $|\mathcal{N}| = 100$; the rest parameter values are set to default as shown in Table IV.

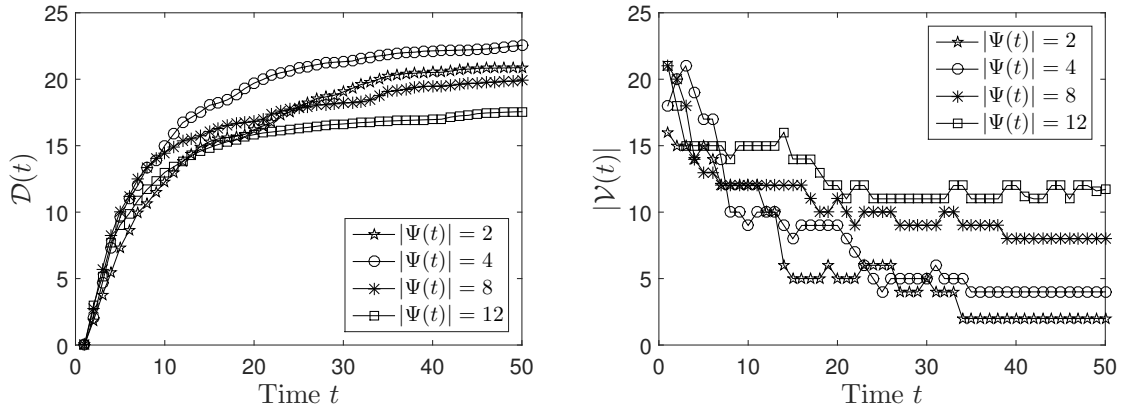

Fig. 18. Scenario III: (left) the traveled distance $\mathcal{D}(t)$ against time t; (right) the number of vanguards $|\mathcal{V}(t)|$ against time t for different number of phenomena $|\Psi| \in \{2, 4, 8, 12\}$ and number of nodes $|\mathcal{N}| = 100$; the rest parameter values are set to default as shown in Table IV.

as depicted by the $\mathcal{A}$ metric in Figure 19 (right). It is worth mentioning that the $|\mathcal{V}(t)|$ metric clearly indicates the merge/split capability of the algorithm. The sudden peaks of $|\mathcal{V}(t)|$ and the increasing trend at the beginning of the intervals $(t_1, t_2]$ and $(t_2, t_3]$ denote that new vanguards are decoupled from existing neighbourhoods (formations) to cover new phenomena. In each interval, the vanguards merge their neighbourhoods, thus, $|\mathcal{V}(t)|$ gets close to the current number of phenomena $|\Psi(t)|$.

Scenario V: probabilistic, multiple phenomena occurrences In this scenario, at $t = 0$, a phenomenon ϕ occurs at a random point. For each time instance $t > 0$ a new phenomenon $\phi(t)$ occurs at a random location $\ell_\phi \in F$ with probability $\alpha \in \{0.02, 0.06, 0.10, 0.14\}$. Figures 21 and 22 show the performance metrics for this scenario with $|\mathcal{N}| = 100$ nodes and the rest parameter values set to default. We can observe the robustness of the algorithm. Specifically, for $\alpha \in \{0.1, 0.14\}$ the nodes cover the 80% and for $\alpha \in \{0.02, 0.06\}$ they cover the 95% of the considered regions, respectively. In addition, the vanguards locate the phenomena quite accurately once $\mathcal{A} < 5$m

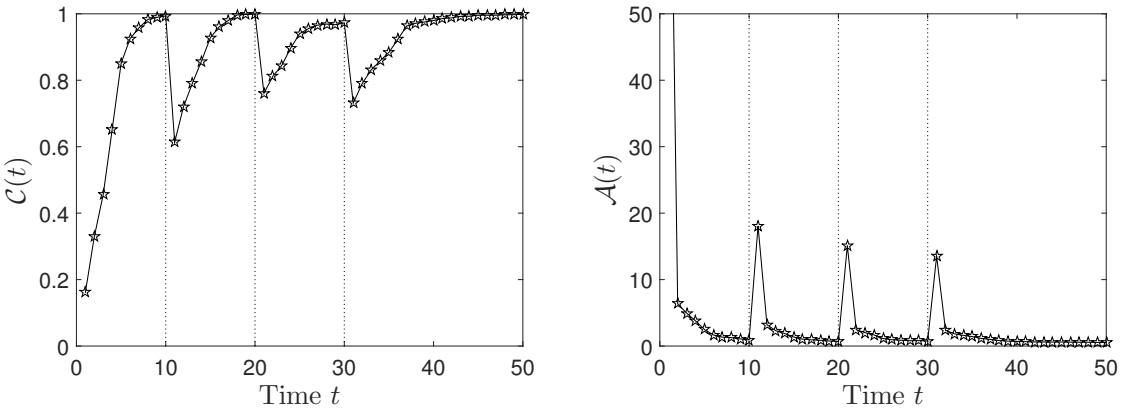

Fig. 19. Scenario IV: (left) the degree of phenomenon coverage $\mathcal{C}(t)$ against time t, (right) the phenomenon closest distance $\mathcal{A}(t)$ against time t with number of nodes $|\mathcal{N}| = 100$; the rest parameter values are set to default as shown in Table IV. The phenomena occur at time instances $t_1 = 10, t_2 = 20$, and $t_3 = 30$.

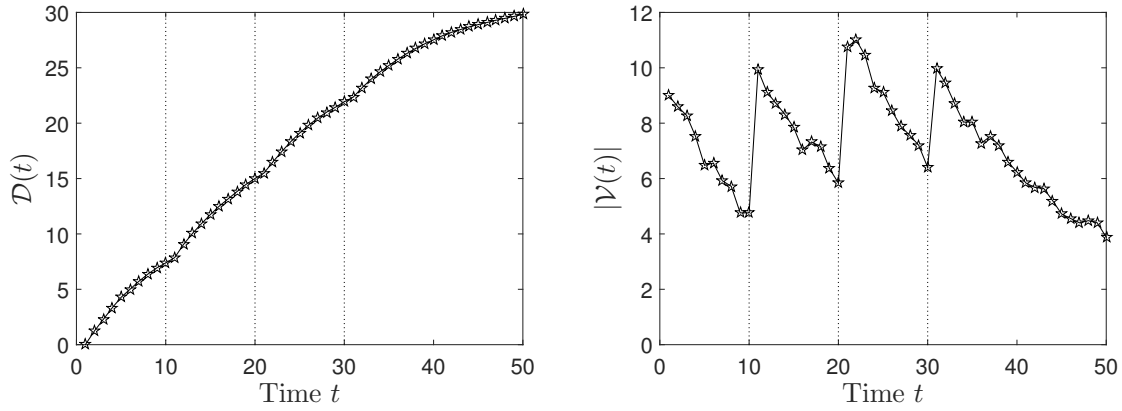

Fig. 20. Scenario IV: (left) the traveled distance $\mathcal{D}(t)$ against time t; (right) the number of vanguards $|\mathcal{V}(t)|$ against time t with number of nodes $|\mathcal{N}| = 100$; the rest parameter values are set to default as shown in Table IV. The phenomena occur at time instances $t_1 = 10, t_2 = 20$, and $t_3 = 30$.

for all probability values. Moreover, we obtain a 40% increase in the travelled distance $\mathcal{D}$ from $\alpha = 0.02$ to $\alpha = 0.14$ (600% increase in α). Finally, the plot of $|\mathcal{V}(t)|$ in Figure 22(right) depicts that the algorithm attempts to reduce the number of vanguards very close to the number of the current phenomena (e.g., from $t = 0$ to $t = 50$ with $\alpha = 0.14$ we have $|\Psi(50)| = 7 + 1$ phenomena while, we obtain $|\mathcal{V}(50)| = 8.7$ vanguards).

We further assess the performance of the algorithm wih respect to the spatial distribution of the sensing nodes. We measure deviations of the spatial distribution from the 'ideal grid' scenario i.e., the scenario where all the available nodes form a perfect grid layout that spans the entire ROI F. We name this metric as **spatial bias indicator** (SBI). The phenomena are generated randomly on the terrain as a Poisson arrival process. The phenomenon duration is also modeled as an exponential random variable, as in Scenario IV with $\sigma = 0.08$. The SBI is quantified through the mean distance of nodes from a fixed point within the sensing field, i.e., point $(0, 0)$. We normalize the mean distance of a certain snapshot of the network through the mean distance of the

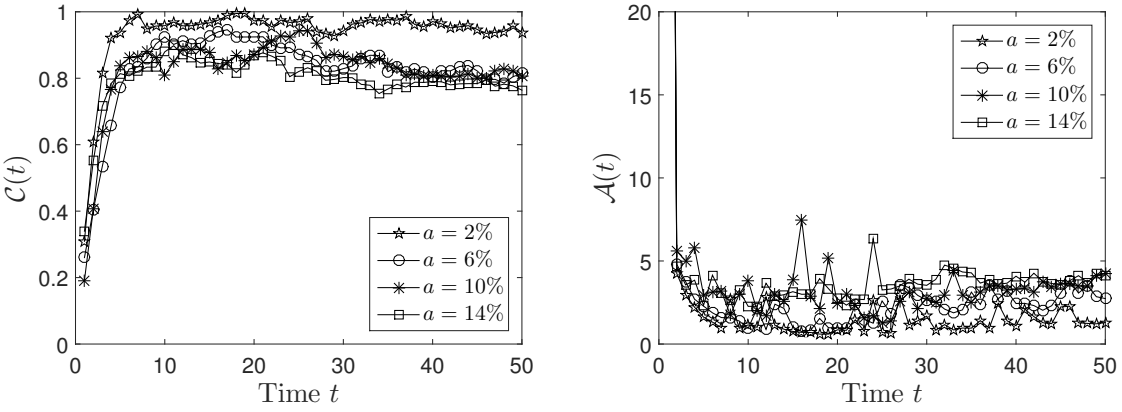

Fig. 21. Scenario V: (left) the degree of phenomenon coverage $\mathcal{C}(t)$ against time t, (right) the phenomenon closest distance $\mathcal{A}(t)$ against time t for different probability of phenomena occurrence $|\alpha| \in \{2, 6, 10, 14\}\%$ and number of nodes $|\mathcal{N}| = 100$; the rest parameter values are set to default as shown in Table IV.

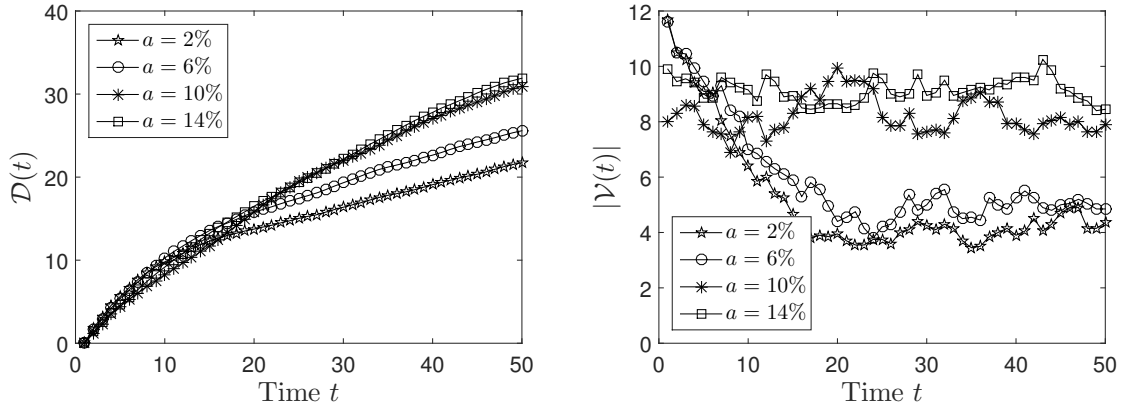

Fig. 22. Scenario V: (left) the traveled distance $\mathcal{D}(t)$ against time t; (right) the number of vanguards $|\mathcal{V}(t)|$ against time t for different probability of phenomena occurrence $|\alpha| \in \{2, 6, 10, 14\}\%$ and number of nodes $|\mathcal{N}| = 100$; the rest parameter values are set to default as shown in Table IV.

'ideal grid' case, i.e., that is

$$SBI(t) = \frac{\sum_{i \in \mathcal{N}} \parallel \ell_i(t) \parallel}{\sum_{i \in \mathcal{N}} \parallel \ell_i(0) \parallel}, t \geq 0.$$

The network of nodes is considered in the ideal grid layout at $t = 0$ and gets 'deformed' as time passes. Therefore, we observe whether the instantaneous SBI values in our simulations differ significantly from unity. Additionally, we assess whether (and how fast) SBI recurs to unity after the occurrence and localization of a phenomenon. In addition, let $\mathcal{Z}(t)$ denote the **percentage degree of coverage of the ROI** F by all nodes at t; $\mathcal{Z}(0)$ indicates the coverage degree of F at time instance $t = 0$ by nodes forming a perfect grid layout. $\mathcal{Z}$ is quantified through Monte Carlo simulation. $Z(t) = \frac{\mathcal{Z}(t)}{\mathcal{Z}(0)}$ is the normalized coverage degree of F. Figure 23 shows the $SBI(t)$ and $Z(t)$ with $|\mathcal{N}| = 100$ and 40 phenomena in total. Such a behaviour shows the capability

of the proposed algorithm to efficiently cover the considered field and keep coverage holes to the lowest possible percentage. In the Figure 23, we observe mean value of $(Z, SBI) = (1.021, 1.005)$.

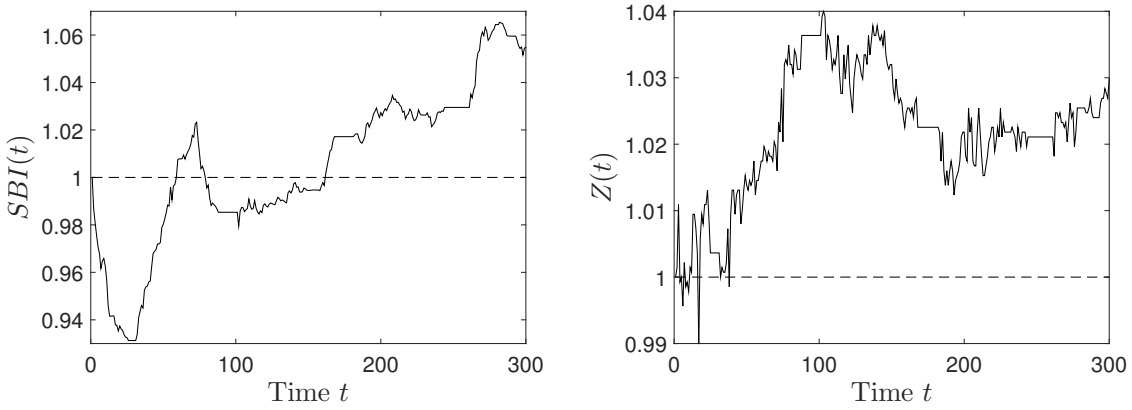

Fig. 23. The SBI and Z metrics. The forty phenomena occur randomly in space & time sensed by $|\mathcal{N}| = 100$ nodes. The rest parameter values are set to default as shown in Table IV.

8.4. Energy Consumption

In this section, we provide comprehensive results of the consumed energy cost per node, which is involved in the distributed phenomena localization process. For each of the above mentioned five scenarios, we report on the total cumulative energy consumption $\mathcal{E}$ in (25), cumulative computation and communication consumption $\mathcal{E}_{RTP} = \mathcal{E}_T + \mathcal{E}_R + \mathcal{E}_P$ and cumulative mobility energy consumption of a node $\mathcal{E}_M$ up to the time horizon that phenomena have been localized by the MSN. During this period, a node is either dynamically appointed as vanguard or acting as a member. We also study the ratio of the total communication and computation energy consumption out of the total mobility energy consumption due to the traveled distance, $\mathcal{E}_{RTP}/\mathcal{E}_M$, and its influence by the number of nodes $|\mathcal{N}|$ in the MSN, the probability of phenomena occurrence α, and the sequential appearance of the phenomena in the sensing field with the time, $|\Psi|$.

For the sensing, communication and computation energy consumption, we adopted the energy model from the Mica2 sensor board [He et al. 2004]. This energy model assumes an energy of two AA batteries that approximately supply 2200 mAh with effective average voltage 3V. It consumes 20mA if running a sensing application continuously, which leads to a lifetime of 100 hours. The communication cost for transmitting a bit is e_{CT} = 720 nJ/bit and receiving a bit is e_{RX} = 110 nJ/bit. The energy consumed by the transceiver power amplifier to transmit one bit over the distance of one meter is e_{TX} = 10^{-3} nJ/m. The path loss exponent y is set to 4, which is usually adopted since path loss exponent hardly achieves the value of 2 in realistic environment [Razzaque & Dobson 2014]. Moreover, the packet header of the communication protocol adopted by Mica2 is 9 bytes (MAC header and CRC) and the maximum payload is 29 bytes. Therefore, the per-packet overhead equals to 23.7% (lowest value). For each transmitted value, e.g., the EP value in an election message, or the location coordinates in a relocation directive message, the assumed payload for a number is set to 4 bytes (floating point number). The maximum communication range for a node is set to $\epsilon = 50$ m

and the coverage radius is $\zeta = 0.35\epsilon$ m, as set in the experiments in Section 8.1; the rest parameter values are set to default as shown in Table IV. Moreover, the movement parameter is configured to $\nu = 1$J/m $= 10^9$ nJ/m assuming a wheeled node (robot) moving on flat concrete terrain at constant friction. According to [Goldenberg et al. 2004] the parameter $\nu = 1$ Joule/m is applicable for a robot node up to weight 10Kg, which is also assumed in our experiments. Some example wheeled robots include Koala 2.5[6] at 4.5Kg or s-bot[7] at 660g. Finally, the energy cost for single CPU instructions (energy per instruction) is 4 nJ/instruction in Mica2 [He et al. 2004]. Table V shows all the energy consumption parameters for this experimental study, while the parameter values corresponding to all sub-processes are set to default as in Table IV.

Table V. Energy parameters

Parameter	**Default values**
Transceiver power amplifier energy consumption, e_{TX}	10^{-3} nJ/m
Transmitting a bit energy consumption, e_{CT}	720 nJ/bit
Receiving a bit energy consumption, e_{RX}	110 nJ/bit
Path loss exponent y	4
CPU energy consumption per instruction	4 nJ/instruction
Movement parameter ν	10^9 nJ/m
Node (robot) weight	10 Kg

For each of the scenarios described in Section 8.3, we study the corresponding cumulative energy consumption (computation/sensing, communication and mobility) as well as the ratio of $\mathcal{E}_{PTR}/\mathcal{E}_M$.

Energy Consumption in Scenario I. Figure 24 shows the cumulative energy $\mathcal{E}$ (left) and the energy $\mathcal{E}_{PTR}$ (right) per node. We can observe that, through the domino-like policy of the algorithm, the mean traveled distance is long with a low number of nodes (thus relatively high mobility energy consumption) and short with a high number of nodes (thus relatively low mobility energy consumption) as also depicted in Figure 11. Specifically, a low number of nodes denotes a low network density and, thus, a relatively high portion of nodes (i) is moving to further cover the area around the vanguard and (ii) is 'continuously' migrated toward the phenomenon. This increases the expected energy consumption per node due to the fact that most of the network nodes move toward the phenomenon POI after each election process. Moreover, due to the low number of nodes, each node is highly probable to be re-elected as a vanguard for many times, thus, increasing the $\mathcal{E}_{PTR}$ energy consumption as shown in Figure 24 (right). On the other hand, with a high number of nodes, which indicates high network density, each node contributes 'temporally' to the relocation process. After that, another nearby node is going to be the next vanguard and so on until reaching the phenomenon. Hence, this denotes that less nodes are re-elected as vanguards, thus apart from traveling short distances, their $\mathcal{E}_{PTR}$ energy consumption decreases as shown in Figure 24 (right). In addition, Figure 25 shows the ratio of the $\mathcal{E}_{PTR}$ consumption with respect to $\mathcal{E}_M$ consumption per node. A low value of this ratio indicates that a node requires more mobility energy to perform its task for the phenomena localization process compared with the computation and communication energy. To this end, a high portion of the energy budget is used for moving around the field area and less energy budget is used for in-network processing (sensing and optimal coverage of areas). The impact of the number of nodes in the energy budget management per node is evidently high, as described above. The trade-off here is the availability of a relatively high number of

[6]http://www.k-team.com/
[7]http://www.swarm-bots.org/

nodes to be deployed onto the sensing field in order to increase the lifetime of the network. Indicatively, with an increase in the number of nodes from fifty to one hundred and fifty (three times more nodes), each node will obtain a 85% decrease in its total energy consumption with a 50% decrease in the ratio of the energy balance $\frac{\mathcal{E}_{PTR}}{\mathcal{E}_M}$.

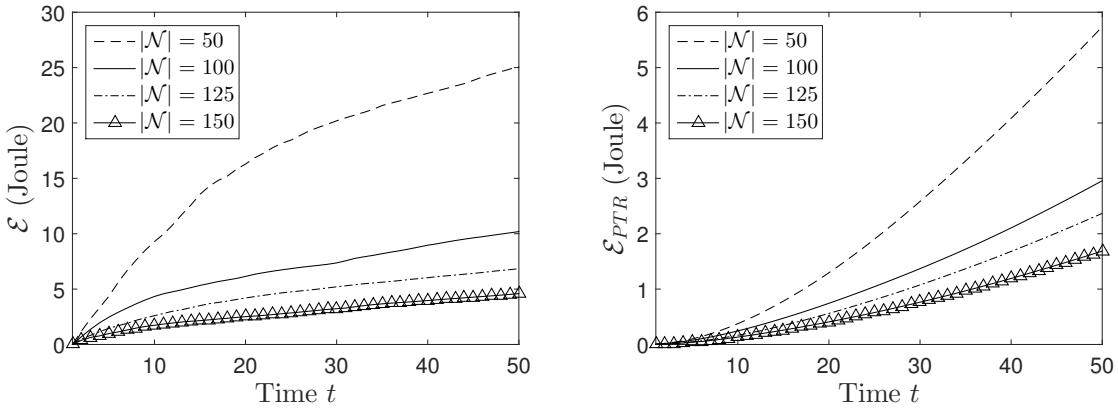

Fig. 24. Scenario I: (left) total cumulative energy consumption $\mathcal{E}$ per node against time t; (right) total cumulative communication and computation energy consumption $\mathcal{E}_{PTR}$ per node for different number of nodes $|\mathcal{N}|$. The rest parameter values are set to default in Table IV and in Table V.

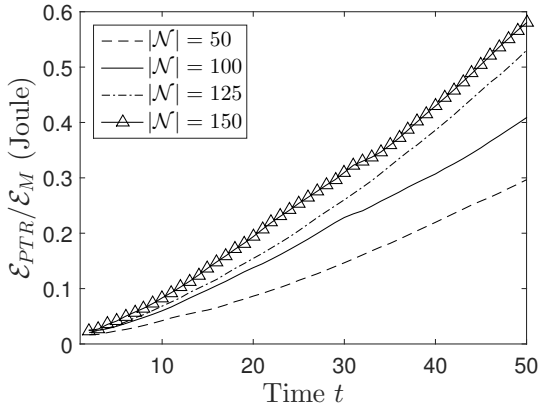

Fig. 25. Scenario I: energy ratio $\mathcal{E}_{PTR}/\mathcal{E}_M$ of computation and communication energy out of the mobility energy per node for different number of nodes $|\mathcal{N}|$. The rest parameter values are set to default in Table IV and in Table V.

Energy Consumption in Scenario II. Similar results on the energy consumption we observe in Scenario II (with these in Scenario I), where the phenomenon under consideration is outside the sensing area F, as shown in Figures 26 and 27. Evidently, in this case, the nodes require more mobility energy since they gradually have to move to a relatively far phenomenon. However, when the nodes approach the phenomenon POI then we obtain similar behavior on the consuming energy as described above for the Scenario I.

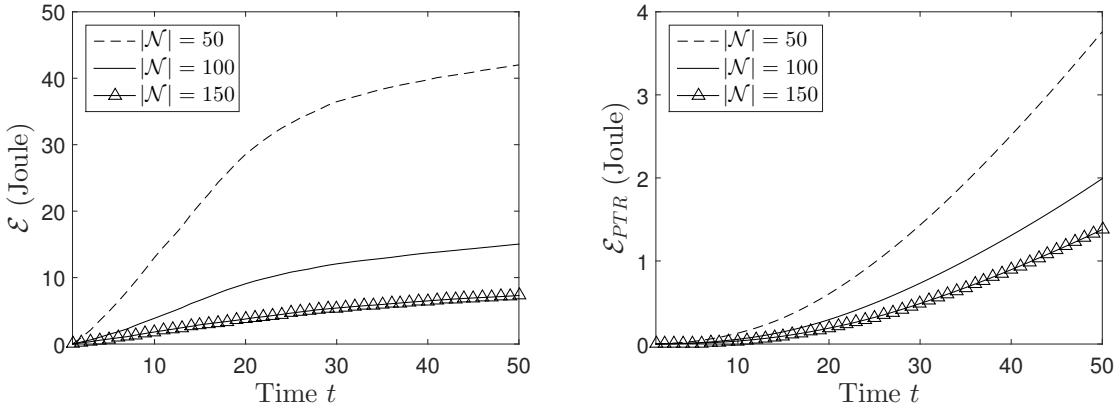

Fig. 26. Scenario II: (left) total cumulative energy consumption $\mathcal{E}$ per node against time t; (right) total cumulative communication and computation energy consumption $\mathcal{E}_{PTR}$ per node for different number of nodes $|\mathcal{N}|$. The rest parameter values are set to default in Table IV and in Table V.

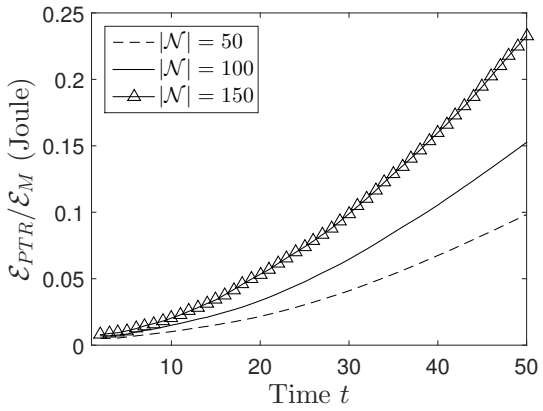

Fig. 27. Scenario II: energy ratio $\mathcal{E}_{PTR}/\mathcal{E}_M$ of computation and communication energy out of the mobility energy per node for different number of nodes $|\mathcal{N}|$. The rest parameter values are set to default in Table IV and in Table V.

Energy Consumption in Scenario III. In Figures 28 and 29, we observe the impact of the number of simultaneous phenomena in a ROI with $|\mathcal{N}| = 100$ nodes on the consumed energy per node. It is worth noting that, the proposed algorithm exhibits a robust behavior in energy consumption with respect of different number of phenomena. As shown in Figure 28 (left), the total energy per node is quite the same when different number of phenomena appear in the ROI. This is attributed to the fact that, the more phenomena occur in a ROI, the more vanguards are elected to simultaneously relocate their members towards the phenomena, as shown in Figure 18 (right). To this end, t he entire MSN is 'separated' into certain swarms each one dedicated to localize a specific phenomenon. Note that, in this case, the $\mathcal{E}_{PTR}$ increases with the increase in the number of phenomena as shown in Figure 28 (right). Specifically, when the simultaneous phenomena increases from two to twelve, then we obtain five times more vanguards during the localization process; a linear increase in $|\Phi|$ results to a linear increase in the number of vanguards $|\mathcal{V}|$. This also results in a 66% increase in the to-

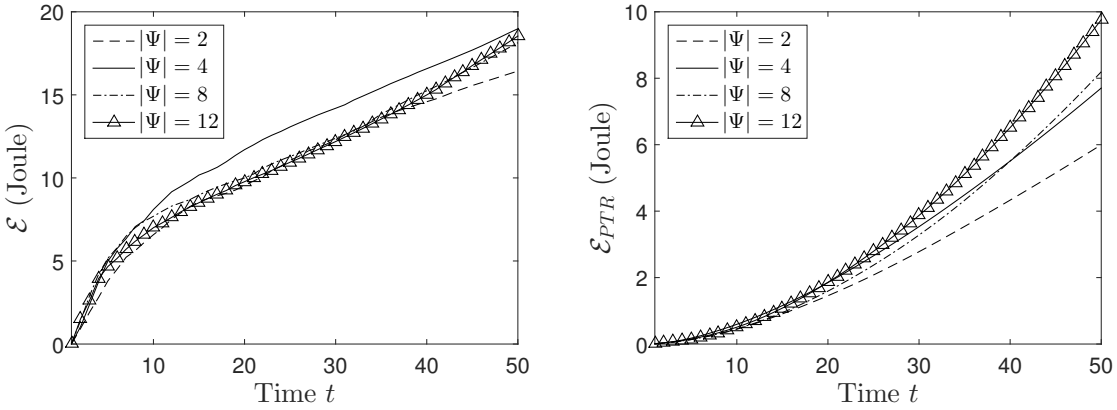

Fig. 28. Scenario III: (left) total cumulative energy consumption $\mathcal{E}$ per node against time t; (right) total cumulative communication and computation energy consumption $\mathcal{E}_{PTR}$ per node for different number of phenomena $|\Psi|$ and number of nodes $|\mathcal{N}| = 100$. The rest parameter values are set to default in Table IV and in Table V.

tal $\mathcal{E}_{PTR}$ consumption per node (i.e., a high portion of the network nodes are appointed as vanguards to localize the multiple phenomena). Moreover, the energy budget balance is almost equally shared to communication/computation and mobility energy, as shown in Figure 29, with the increase of the number of phenomena. Notably, when $|\Psi| = 12$, we obtain $\mathcal{E}_{PTR} \simeq \mathcal{E}_M$. This denotes that the MSN has appointed a relatively high number of vanguards (specifically, on average 12.6% of the nodes are vanguards), thus, increasing the communication and computation energy consumption on nodes which refers to the same energy budget required for mobility. On the other hand, a low number of phenomena results to a low number of vanguards, thus, the $\mathcal{E}_{PTR}$ per node is lower compared with the case of a high number of vanguards. This implies a lower ratio $\frac{\mathcal{E}_{PTR}}{\mathcal{E}_M}$, as shown in Figure 29.

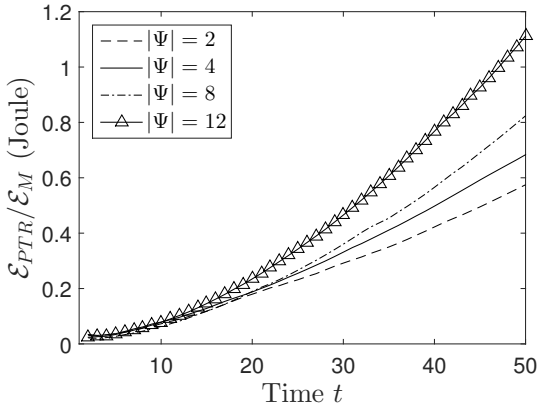

Fig. 29. Scenario III: energy ratio $\mathcal{E}_{PTR}/\mathcal{E}_M$ of computation and communication energy out of the mobility energy per node for different number of phenomena $|\Psi|$ and number of nodes $|\mathcal{N}| = 100$. The rest parameter values are set to default in Table IV and in Table V.

Energy Consumption in Scenario IV. Figure 30 shows the total consumed energy per node in Scenario IV for $|\mathcal{N}| = 100$ nodes in which sequential phenomena occur with a specific lifetime duration. The results are quite similar with those of Scenarios I and II in terms of energy consumption and ratio between $\mathcal{E}_{PTR}/\mathcal{E}_M$. We can observe that a very low portion of the $\mathcal{E}_{PTR}$ energy is consumed compared to the mobility cost $\mathcal{E}_M$. This is due to the fact that, nodes when directing to a sensed phenomenon through the domino-like policy, are subject to split because a new phenomenon occurs. Consequently, when another phenomenon is captured, then, again, some neighborhoods are split or new nodes start moving towards to the newly captured phenomenon and so on. Hence, there is a portion of nodes that almost continuously moves in the area with the purpose of localizing the sequentially generating phenomena.

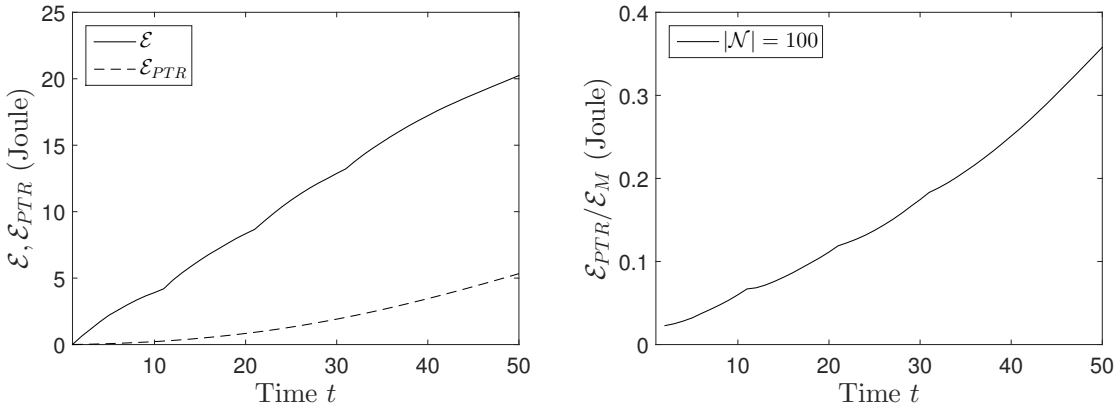

Fig. 30. Scenario IV: (left) total cumulative energy consumption $\mathcal{E}$ per node and total cumulative communication and computation energy consumption $\mathcal{E}_{PTR}$ per node against time t; (right) energy ratio $\mathcal{E}_{PTR}/\mathcal{E}_M$ of computation and communication energy out of the mobility energy per node with number of nodes $|\mathcal{N}| = 100$. The rest parameter values are set to default in Table IV and in Table V.

Energy Consumption in Scenario V. Figure 31 shows the consumed energy per node for $|\mathcal{N}| = 100$ nodes in Scenario V, in which phenomena occur with certain probability of occurrence α. As we observe from Figure 22 (right), the number of vanguards is higher when the phenomena occurrence probability is high in a ROI. Moreover, when new phenomena occur, then more vanguards are elected to 'localize' their POIs. This evidently increases both the consumed mobility energy and the computation and communication energy. Indicatively, a six times increase in the probability of occurrence α results in a 70% increase in the consumed $\mathcal{E}_{PTR}$ energy due to the fact that more vanguards are needed to communicate with their neighbors and locally execute the CPSO algorithm. After that, more nodes are being relocated as dictated by their vanguards to localize all the randomly generating phenomena. It is worth noting that, our algorithm is robust in balancing the energy consumption between mobility and computation/communication with a constant ratio of 0.4 as the probability of occurrence increases from $\alpha = 2\%$ to $\alpha = 14\%$. As shown in Figure 32, an increase in α results into an 'constant' increase in both energy budgets ($\mathcal{E}_{PTR}$ and $\mathcal{E}_M$) with node mobility requiring more energy budget than computation and communication.

9. DISCUSSION ON THE ROI COVERAGE PROBLEM

The PSO algorithm has gained increasing popularity among researchers in various domains [Cai et al. 2009, Cui and Turan 2010, Montalvo et al. 2010]. In our setting,

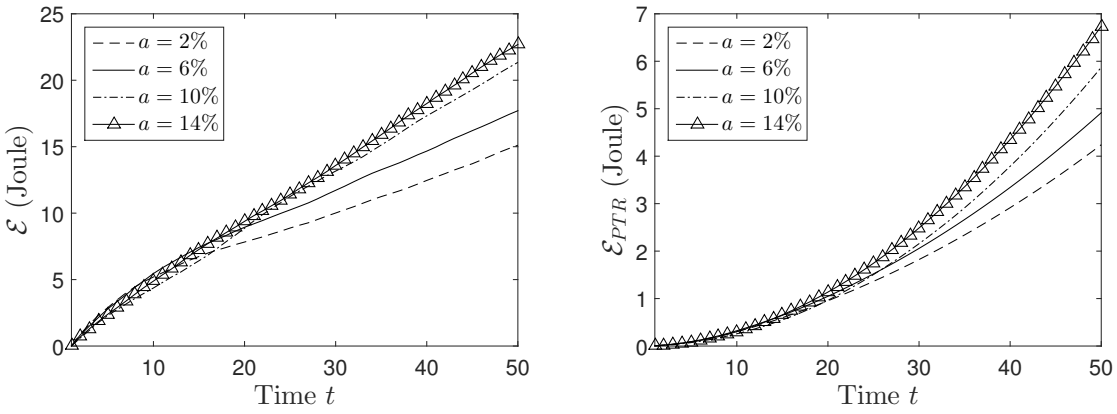

Fig. 31. Scenario V: (left) total cumulative energy consumption $\mathcal{E}$ per node against time t; (right) total cumulative communication and computation energy consumption $\mathcal{E}_{PTR}$ per node for different probability of phenomena occurrence $\alpha \in \{2, 6, 10, 14\}\%$ and number of nodes $|\mathcal{N}| = 100$. The rest parameter values are set to default in Table IV and in Table V.

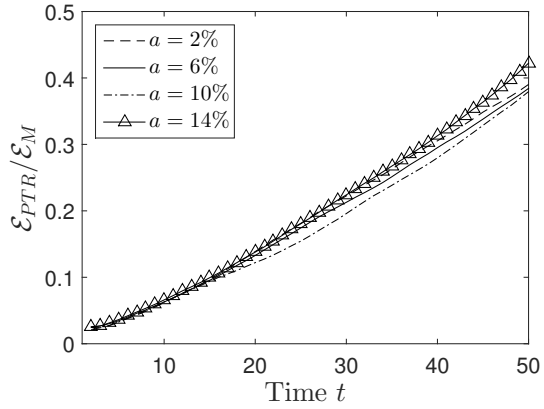

Fig. 32. Scenario V: energy ratio $\mathcal{E}_{PTR}/\mathcal{E}_M$ of computation and communication energy out of the mobility energy per node for different probability of phenomena occurrence $\alpha \in \{2, 6, 10, 14\}\%$ and number of nodes $|\mathcal{N}| = 100$.The rest parameter values are set to default in Table IV and in Table V.

CPSO (a variant of the PSO algorithm) is adopted by certain nodes (vanguards) for solving the ROI coverage problem as discussed in Section 4. PSO is considered as a robust and efficient technique for solving population-based stochastic optimization problems. In this section, we report on the adoption of the PSO algorithm as a multivariate optimization algorithm applied to the ROI coverage problem.

In PSO, a population of particles begins to move in the search space in order to find the optimal solution. The rationale of adopting the PSO in our problem is that, unlike Genetic Algorithms (GA), it does not need any complex encoding / decoding processes or special operator. Particles follow the current optimum 'teammates' and update their positions in order to finally reach the optimal result. Particles positions are candidate solutions while updates in the positions are observed at every iteration based on the local best, global best, and centroid best (in CPSO) positions. The behaviour of each particle is very simple: *try to find the best solution following the 'success' of your 'team-*

mates'. Hence, all particles tend to reach the final best solution in the search space. However, in some cases, this feature makes the population vulnerable to the possibility of stagnation in a local minimum [Crepnisek et al. 2013]. In GAs, the mutation operation is generally used to bring the population out of a local minimum. The mutation operator has been successfully applied to the particle's inertia weight and acceleration coefficients in PSO [Zhan et al. 2009].

There might be alternative multivariate optimization algorithms that could solve our problem. The selection of a different algorithm does not alter the main idea of our approach for the dynamic phenomena identification, since this algorithm is invoked by a vanguard in order to disseminate RDs to its neighboring nodes for optimally covering a given ROI. However, the requirements of adopting an alternative algorithm is (a) the low execution time of the optimization process run on a vanguard and (b) the locally available spatial information of the vanguard's neighbours. We report on the adoption of the most popular methods originated in Swarm Intelligence: Ant Colony Optimization (ACO) and Artificial Bee Colony (ABC). We qualitatively and quantitatively (through a comparative assessment of CPSO with ABC) provide a discussion on the adoption of ACO and ABC, respectively, to our problem.

9.1. The ACO method

An artificial Ant Colony System (ACS) is an agent-based system, which simulates the natural behaviour of ants and develops mechanisms of cooperation and learning. ACS was proposed in [Dorigo and Cambardella 1997, Dorigo et al. 1999] as a new heuristic to solve combinatorial optimization problems. This heuristic, called Ant Colony Optimization (ACO), has been found to be both robust and versatile in handling a wide range of combinatorial optimization problems. The main idea of ACO is to model a problem as the search for a minimum cost path in a graph. Artificial ants walk on this graph, looking for low cost paths according to a fitness function. Each ant has a rather simple behaviour capable of finding relatively costlier paths. Cheaper paths are found as the result of the global cooperation among ants. The behaviour of the artificial ants is inspired from real ants: *they lay pheromone trails on the graph edges and choose their path with respect to probabilities that depend on the pheromone trails*. These pheromone trails progressively decrease by evaporation. Artificial ants live in a discrete world (i.e., the graph) and their movements consist of transitions from vertices to vertices. The ACO algorithm is proved efficient for the Traveling Salesman Problem and, normally, solves similar discrete optimization problems [Dorigo and Cambardella 1997]. The ACO is more applicable for problems where source and destination are predefined and specific. The adoption of the ACO algorithm would not be successful for solving our problem. In our case, we attempt to find a solution within a continuous search space (i.e., coordinates of the nodes in ROI F) and not to find an optimal path between positions in a discrete, graph-represented search space, where the destination is unknown.

9.2. The ABC method

In the Artificial Bee Colony (ABC) algorithm [Karaboga 2005, Karaboga and Akay 2009], the colony of artificial bees contains three groups of bees: *employed*, *onlookers* and *scouts*. A food source for bees represents a possible solution to the optimization problem. The quality of the solution is depicted by the amount of nectar (fitness value) that a food source corresponds to. A bee, which directs to the food source visited previously by itself, refers to as an employed bee. A bee, which waits on the 'dance' area for choosing a food source, is called an onlooker. A bee, which is responsible for carrying out random search, is called a scout. For every food source, there is only one employed bee. The employed bee, whose food source is exhausted by the employed and onlooker

bees, becomes a scout. Initially, a set of food sources are randomly selected by the bees and their nectar amounts are determined. The number of the employed bees (or the onlooker bees) is equal to the number of solutions in the population. Each iteration of the ABC algorithm consists of three steps: (i) send the employed bees onto the food sources and, then, measure their nectar amounts; (ii) select the food sources by the onlookers after sharing the information of employed bees and determine the nectar amount, (iii) determine the scout bees and, then, send them onto the possible food sources. As the nectar amount of a food source increases, the probability that this food source will be chosen by an onlooker increases, too. Hence, the employed bees carrying higher nectar amounts recruit the onlookers for the food source areas corresponding to high nectar amounts.

9.2.1. Comparative assessment. The ABC algorithm controls the foraging behaviour of the bees through the following parameters: (i) the bee colony size M, which is the number of employed bees and the number of onlooker bees, (ii) the number of food sources, which corresponds to the population size (number of particles) m in CPSO; the number of sources is half of the colony size M, i.e., $m = 0.5M$, (iii) the dimension of the food source $D = 2n$, which corresponds to the particle dimension in CPSO, and (iii) the maximum number of iterations, which refers to the T parameter in CPSO. Compared with the PSO, the ABC exhibits better convergence (w.r.t. fitness value), however, this convergence could be poorer when applied in constrained problems, composite functions and some non-separable functions [Cai et al. 2009]. In such cases, the ABC algorithm convergences more slowly compared to other swarm intelligence techniques like PSO [Cai et al. 2009]. Moreover, the ABC algorithm adopts random initialization of the population leading to solutions often concentrated in a local area. The ABC algorithm is efficient in solution exploration but poor at exploitation for two reasons [Cui and Turan 2010]: (a) the coefficient affecting the selection of every food source is random enough for exploration [Montalvo et al. 2010], (b) the ABC is based on a greedy selection mechanism between the old and new solutions, a characteristic that makes the algorithm to be trapped in a local optimal. ABC enforces higher computational requirements for reaching the final solution compared to the PSO, as it will be shown in the following.

We report on a comparative assessment of the CPSO and ABC algorithms for the $F(\ell, \zeta)$ coverage problem. The ABC algorithm is provided in the Appendix. In order to objectively compare the performance of both algorithms, we set the same maximum number of iterations for convergence, the same dimension of the search space (i.e., dimension of the particle in CPSO and dimension of the food source in ABC), and the same population size, i.e., number of particles in CPSO and number of food sources/employed bees in ABC. We study the convergence fitness value of $J(\mathbf{p})$ of the optimal solution $\mathbf{p}$ and the overall execution time T^{all} measured in milliseconds for convergence. The simulation parameters for the comparative assessment for both algorithms is shown in Table VI.

In Table VII, we show the convergence $F(\mathbf{p})$ value of the ABC and CPSO algorithms for certain m and n values having $\zeta = 0.35\epsilon$; similar results are obtained for other values of ζ. We observe that both algorithms assume very similar convergence values for all m and n values, with the ABC algorithm being slightly better than CPSO. Hence, the ABC algorithm can be a possible candidate optimization algorithm for the ROI coverage problem by achieving very high degrees of coverage. However, we have to examine the computational effort of ABC (in terms of execution time), which is deemed a crucial factor in our problem. Figure 33 illustrates the overall execution time T^{all} of both ABC and CPSO for the same maximum number of iterations T. Obviously, the CPSO algorithm requires more or less half of the execution time of the ABC algorithm. This is attributed to the fact that, the ABC algorithm performs more complex compu-

Table VI. The parameters for the comparative assessment of CPSO and ABC in the $F(\ell, \zeta)$ coverage problem.

Parameter	Description	Value
n	Number of nodes within $F(\ell, \epsilon)$; $D = 2n$ is the particle dimension in CPSO and the food source dimension in ABC.	$\{20, 40, 60\}$
m	Population size; i.e., number of particles / employed bees. In ABC, the colony size $M = 2m$ i.e., employed bees plus onlookers	$\{10, \ldots, 60\}$
ζ	Radius of the coverage ROI $F(\ell, \zeta)$.	0.35ϵ
ϵ	Communication range.	50m
δ	Sensing range.	3m
T	Maximum number of iterations.	100

tations in order to lead the bee colony to food sources with high amount of nectar, thus, employing both onlookers and employed bees. ABC requires a double colony size compared to CPSO, which the latter deals only with the size of the particles. On the other hand, ABC achieves slightly higher fitness values compared to CPSO, but, nevertheless, this higher fitness values is obtained with duplicating the population size.

Since both optimization algorithms are applicable to our problem, we could adopt a hybrid method of invoking the CPSO and ABC. Since, CPSO performs better than ABC in terms of execution time, then at the beginning of the sensor field exploration, vanguards could invoke the CPSO algorithm for a first placement of nodes in the sensor field. As soon as phenomena are detected, then, for a more accurate detection results, some nodes, e.g., those which receive high magnitude volumes of the current phenomena, could perform ROI coverage through the invocation of the ABC algorithm, thus, better exploring the closest areas to the phenomena with high degree of coverage.

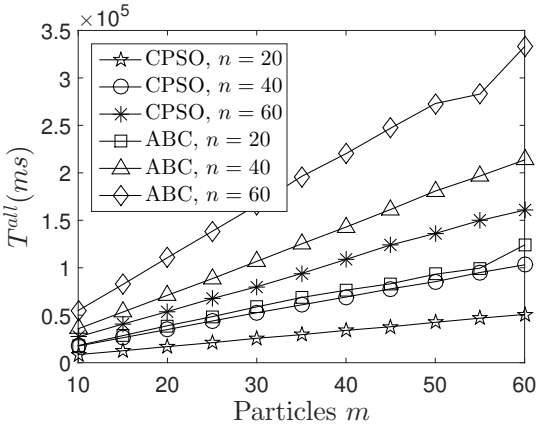

Fig. 33. The total execution time T^{all} in ms. of CPSO and ABC against population size m with different n values; $\zeta = 0.35\epsilon$.

10. CONCLUSIONS AND FUTURE WORK

We present a distributed self-reorganization and relocation process for localizing phenomena (e.g., fire events) by mobile sensor nodes. We treat phenomena through POIs and the magnitude of a certain environmental parameter (e.g., temperature). Such

Table VII. The convergence $J(\mathbf{p})$ value with $\zeta = 0.35\epsilon$ for ABC and CPSO (in parentheses).

	Population size m					
# Nodes n	**10**	**20**	**30**	**40**	**50**	**60**
20	.97(.96)	.98(.97)	.99(.99)	1.0(.99)	1.0(.99)	1.0(1.0)
40	.98(.98)	.99(.99)	1.0(.98)	1.0(1.0)	1.0(1.0)	1.0(1.0)
60	1.0(.99)	1.0(1.0)	1.0(.99)	1.0(1.0)	1.0(1.0)	1.0(1.0)

magnitude is spatially dependent similarly to the potential of the electric field. The proposed process, which applies a specific PSO technique, the Centroid PSO (CPSO), is purely distributed and incrementally executed by vanguards. We evaluate the coverage and localization performance of the proposed process through simulations. Moreover, we conducted a comprehensive experimental evaluation on the consumed energy of our process. Our findings show that the proposed process not only captures a certain phenomenon with high accuracy but also reacts promptly and optimally manages the available sensing resources in case of multiple random occurrences of phenomena. In addition, our algorithm demonstrates a very good energy expenditure as it requires minimum node relocation (between the current and the future position of mobile sensor nodes). Furthermore, we study the adoption of other optimization algorithms from the Computational Intelligence, i.e., the Artificial Bees Colony (ABC), for solving our problem. We conclude that the ABC algorithm can be invoked by the vanguards for better exploration of the areas close to the phenomena, while the CPSO algorithm performs better exploration of a wider area since it requires significantly less execution time than ABC.

In our future research agenda, we investigate more intelligent obstacle-aware relocation directives issued by vanguards for either gathering nodes or evacuating a certain ROI according to the dynamics of the monitored phenomena. Moreover, we study the capability of the vanguard to control the neighborhood splits due to the constraint of maintain a sufficient number of nodes for optimally covering the required ROI. In addition, the proposed algorithm has to be enhanced by dealing with the presence of uncertainty, disturbance and measurement errors. Specifically, nodes determine locally the received magnitude value in order to reason about the occurrence of a phenomenon, thus, the inherent uncertainty on that measurement might lead to a misleading election as a vanguard. A more sophisticated election algorithm, which takes into account a degree of uncertainty on the measurement of the magnitude value and infers whether a node is elected as a vanguard under uncertainty is another issue for future research. Moreover, vanguards base the relocation directives on their measurements and, more importantly, an event of a neighborhood split could be badly initiated by measurement error. It is of high importance to deal with such uncertainty, which effects the robustness of the proposed scheme. In this direction, approximate reasoning algorithms and inference under vague knowledge, e.g., through Fuzzy Set theory and Fuzzy Logic, could be studied as candidate techniques.

Acknowledgment

This work has been co-financed by the European Union (European Social Fund–ESF) and Greek national funds through the Operational Program 'Education and Lifelong Learning' of the National Strategic Reference Framework (NSRF) in the scope of the Research Funding Program: THALES–UOA–Sensor Web Fire Shield (SWeFS).

REFERENCES

Adams, T.M. 'G104-A2LA Guide for estimation of measurement uncertainty in testing', American Association for Laboratory Accreditation: Frederick, MD, USA; July, 2002.

Akcan, H., Kriakov, V., Bronnimann, H., Delis, A., 'Managing Cohort Movement of Mobile Sensors via GPS-free & Compass-free Node Localization', Journal of Parallel and Distributed Computing, 70(7): 743–757, 2010.

Anagnostopoulos, C., Hadjiefthymiades, S., 'Enhancing Situation-Aware Systems through Imprecise Reasoning', IEEE Transactions on Mobile Computing, 7(10):1153–1168, Oct. 2008

Averill M. L., David Kelton, W., 'Simulation Modeling and Analysis', 3rd, McGraw–Hill, 0070592926, 2000.

Azlina, N., Aziz, A., Mohemmed, A. W., Zhang, M., 'Particle Swarm Optimization for Coverage Maximization and Energy Conservation in Wireless Sensor Networks', LNCS 6025, pp. 51–60, Applications of Evolutionary Computation, 2010.

Bai, X., Xuan, D., Yun, Z., Lai, T. H., Jia, W., 'Complete optimal deployment patterns for full-coverage and k-connectivity ($k \leq 6$) wireless sensor networks', 9th ACM MobiHoc08, pp.401–410, 2008.

Bartolini N., Calamoneri T., Fusco E., Massini A., Silvestri S., 'Snap and Spread: A Self-deployment Algorithm for Mobile Sensor Networks', Distributed Computing in Sensor Systems, LNCS 5067, pp. 451–456, 2008.

Bulusu, N., Heidemann, J., Estrin, D., Tran, T., 'Self-configuring localization systems: Design and experimental evaluation', ACM Trans. Embed. Comput. Syst. 2004, 3, 24–60.

Cai, X., Cui, Y., Tan, Y., 'Predicted modified PSO with time-varying Accelerator coefficients', Int. J. of Bio-Inspired Computation, 1(1/2):50–60, 2009.

Chakrabarty, K., Iyengar, S., Qi, H., Cho, E., 'Coding theory framework for target location in distributed sensor networks', In Proc. International Symposium on Information Technology: Coding and Computing, pp.130–134, 2001.

Chakrabarty, K., Iyengar, S., Qi, H., Cho, E., 'Grid coverage for surveillance and target location in distributed sensor networks', IEEE Trans. Comput. 51(12):1448–1453, 2002.

Chandy, K. M., Bunn, J. J., Liu, A. H., 'Models and Algorithm for Radiation Detection' Workshop on Grand Challenges in Modeling, Simulation, and Analysis for Homeland Security (MSAHS), Washington DC, 2010.

Cheng, R., Jin, Y. 'A Social Learning Particle Swarm Optimization Algorithm for Scalable Optimization', Information Sciences, 291: 43–60, 2015.

Cheng, R., Jin, Y. 'A Competitive Swarm Optimizer for Large Scale Optimization', IEEE Transactions on Cybernetics, 45(2): 191–204, 2015.

Cook, M., Franceschetti, M., Bruck, J., 'A geometric theorem for approximate disk covering algorithms', TR, California Inst. of Technology 2001.

Cortes J., Martinez S., Karatas T., Bullo F., 'Coverage control for mobile sensing networks', IEEE Transactions on Robotics and Automation, 20(2):243–255, April 2004.

Crepinsek, M., Liu, S.-H., Mernik, M., 'Exploration and Exploitation in Evolutionary Algorithms: A Survey', ACM Comput. Surv., 45(3):35:1–35:33, 2013.

Cui, H., Turan, O., 'Application of a new multi-agent hybrid co-evolution based particle swarm optimisation methodology in ship design', Computer-Aided Design, 42(11):1013–1027, 2010.

de A Campello, A.C., Jorge, G.C., Costa, S.I.R., 'Decoding q-ary lattices in the Lee metric', Information Theory Workshop (ITW), 2011 IEEE , pp.220–224, 2011

Deb, K., Padhye, N. 'Improving a Particle Swarm Optimization Algorithm Using an Evolutionary Algorithm Framework', KanGAL Tech. Report No.2010003, Feb. 2010.

Dhillon, S., Chakrabarty, K., 'Sensor placement for effective coverage and surveillance in distributed sensor networks', IEEE Wireless Communications and Networking, vol. 3, pp. 1609–1614, March 2003.

Dorigo, M., Di Caro, G., Gambardella, L. M., 'Ant Algorithms for Discrete Optimization', Artificial Life, 5(2):137–172, 1999.

Dorigo, M., Gambardella, L. M., 'Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem', IEEE Transactions on Evolutionary Computation, 1(1):53–66, 1997.

Eberhart, R. C., Kennedy, J., 'A new optimizer using particle swarm theory', 6th Symposium on Micro Machine and Human Science, pp. 39–43, 1995.

Eberhart, R. C., Shi, Y. H., 'Particle swarm optimization: Developments, applications and resources', IEEE Congress on Evolutionary Computation, vol. 1, pp. 81–86, 2001.

Elfes, A., 'Occupancy grids: A stochastic spatial representation for active robot perception', In Proc. 6th Conference on Uncertainty in AI, pp.60–70, 1990.

Esnaashari, M., Meybodi, M.R., 'A cellular learning automata-based deployment strategy for mobile wireless sensor networks', Journal of Parallel and Distributed Computing, 71(7):988–1001, 2011.

Gaojun Fan, Shiyao Jin, 'Coverage Problem in Wireless Sensor Network: A Survey', Journal of Networks, Academy Publisher, 5(9):1033–1040, Sept. 2010.

Garetto, M., Gribaudo, M., Chiasserini, C., Leonardi, E., 'A distributed sensor relocation scheme for environmental control', 4th IEEE Intl. Conf. on Mobile Ad-hoc and Sensor Systems (MASS 2007), 2007.

Gifford, C. M., Akers, E.L., Stansbury, R.S., Agah, A. 'Mobile Robots for Polar Remote Sensing', In Guarav S. Sukhatme (Ed.), The Path to Autonomous Robots, Springer-Verlag, Heidelberg, Germany, pp. 3–24, Feb. 2009.

Gifford, C.M., Finyom, G., Jefferson, M., Reid, M., Akers, E. L., Agah, A. 'Automated Polar Ice Thickness Estimation from Radar Imagery', IEEE Transactions on Image Processing, 19(9):2456–2469, Sept. 2010.

Goldenberg, D., Lin, J., Morse, A., Rosen, B., Yang, R., 'Towards mobility as a network control primitive', In Proceedings of the 5th ACM international symposium on Mobile ad hoc networking and computing (MobiHoc '04). ACM, New York, NY, USA, 163–174, 2004.

Golomb, S., Welch, L., 'Perfect codes in the lee metric and the packing of polyominoes', SIAM J. Appl Math., 18:302–317, 1970

Huang, C-F., Tseng, Y-C., 'The coverage problem in a wireless sensor network', Mob. Netw. Appl., 10(4):519–528, August 2005

Helm, I., Jalukse, L., Leito, I. 'Measurement uncertainty estimation in amperometric sensors: a tutorial review', Sensors (Basel), 10(5):4430–55, Apr 2010.

Heo, N., Varshney, P.K., 'Energy-efficient deployment of Intelligent Mobile sensor networks, IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, 35(1):78–92, Jan. 2005.

Hernandez, V. B., Lilienthal, A.J., Neumann, P., Trincavelli, M. 'Mobile robots for localizing gas emission sources on landfill sites: is bio-inspiration the way to go?' Frontiers in Neuroengineering, 4(0), 2012.

Hernandez, V.B., Schaffernicht, E., Pomareda. V., Lilienthal, A.J., Marco, S., Trincavelli, M. 'Combining Non Selective Gas Sensors on a Mobile Robot for Identification and Mapping of Multiple Chemical Compounds', Sensors, (Basel, Switzerland), 14(9):17331–17352, 2014.

Kar, K., Banerjee, S., 'Node Placement for Connected Coverage in Sensor Networks', In WiOpt'03: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks, 2003.

Karaboga, D., 'An Idea Based On Honey Bee Swarm for Numerical Optimization', Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.

Karaboga, D., Akay, B., 'A comparative study of Artificial Bee Colony algorithm', Applied Mathematics and Computation, 214:108–132, 2009.

Kennedy, J., Eberhart, R. C., 'Particle swarm optimization', IEEE Conference on Neural Networks,4:1942–1948, 1995.

Kulkarni, R. V., Venayagamoorthy, G. K., 'Particle Swarm Optimization in Wireless Sensor Networks: A Brief Survey', IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, 41(2):262–267, March 2011.

Kumar, N., Gunopulos, D., Kalogeraki, V., 'Sensor Network Coverage Restoration', V. Prasanna et al. (Eds.), Im DCOSS 2005, LNCS 3560, p. 409, 2005.

Labella, T. H., Fuchs, G., Dressler, F.,'A simulation model for self-organized management and sensor/actuator networks', GI/ITG KuVS Fachgesprch Selbstorganisierende, Adaptive, Kontextsensitive verteilte Systeme, SAKS 2006.

Li X., Frey H., Santoro N., Stojmenovic I., 'Strictly localized sensor self-deployment for optimal focused coverage', IEEE Transactions on Mobile Computing, no.99, Dec. 2010, 10.1109/TMC.2010.261.

Li, X., Nayak, A., Simplot-Ryl, D., Stojmenovic, I., 'Sensor Placement in Sensor and Actuator Networks', Wireless Sensor and Actuator Networks, John Wiley & Sons, Inc., 263–294, 2010.

Li, Y-L., Shao,W., You, L., Wang, B-Z. 'An Improved PSO Algorithm and Its Application to UWB Antenna Design', Antennas and Wireless Propagation Letters, IEEE , vol. 12, pp.1236–1239, 2013.

Liang, S., Song, S., Kong, L., Cheng, J., 'An Improved Particle Swarm Optimization Algorithm and its Convergence Analysis', 2nd Intl. Conf. Computer Modeling and Simulation (ICCMS '10), vol.2, pp.138–141, 2010.

Manolakos, E., Manatakis, D., Xanthopoulos, G., 'Temperature field modelling and simulation of wireless sensor network behaviour during a spreading wildfire', European Signal Processing Conference (EUSIPCO 2008), August 2008.

Marengoni, M., Draper, B. A., Hanson, A., Sitaraman, R. A., 'A System to Place Observers on a Polyhedral Terrain in Polynomial Time', Image and Vision Computing, 18:773, De. 1996.

Megerian S., Koushanfar F., Potkonjak M., Srivastava M.B., 'Worst and best-case coverage in sensor networks', IEEE Transactions on Mobile Computing, 4(1):84–92, Jan.–Feb. 2005.

Meguerdichian, S., Koushanfar, F., Potkonjak, M., Srivastava, M. B., 'Coverage problems in wireless ad-hoc sensor networks', IEEE INFOCOM 2001, vol.3, pp. 1380–1387, 2001.

F. Mondada, G. C. Pettinaro, A. Guignard, I. W. Kwee, D. Floreano, J-L Deneubourg, S. Nolfi, L. M. Gambardella, M. Dorigo, 'Swarm-Bot: A New Distributed Robotic Concept', Autonomous Robots, 17(2-3):193–221, Sept. 2004.

Montalvo, I., Izquierdo, J., Perez–Garcia, R., Herrera, M., 'Improved performance of PSO with self-adaptive parameters for computing the optimal design of water supply systems', Engineering Applications of Artificial Intelligence, 23(5):727–735, Aug. 2010.

Paterson, M., Fowler, R., Tanimoto, S., 'Optimal packing and covering in the plane are NP complete', Information Processing Letters, 12(3):133–137, 1981.

Pan, D., Liu, Z. 'An Improved Particle Swarm Optimization Algorithm', Emerging Research in Artificial Intelligence and Computational Intelligence, Communications in Computer and Information Science, Vol. 237, pp. 550–556, 2011.

Razzaque, M., Dobson, S., 'Energy-Efficient Sensing in Wireless Sensor Networks Using Compressed Sensing', Sensors 2014, 14, 2822–2859.

Savvides, A., Han, C.C.l Srivastava, M. B.l 'Dynamic Fine Grained Localization in Ad-Hoc Sensor Networks', MobiCom'01, pp. 166–179, 2001.

Sayed-Mouchaweh, M., Lughofer, E., 'Learning in Non-Stationary Environments: Methods and Applications', DOI 10.1007/978-1-4419-8020-5 4, Springer Science+Business Media, New York, 2012.

Sekkas, O., Hadjiefthymiades, S., Zervas, E. 'A Multi-level Data Fusion Approach for Early Fire Detection', 1st International Workshop on Computational Intelligence for Disaster Management (CIDM-2010), 2010.

Shi, Y., Eberhart, R. C., 'A modified particle swarm optimizer', IEEE World Congress on Computational Intelligence, pp. 69–73, 1998.

Shi, Y., Eberhart, R. C., 'Empirical study of particle swarm optimization', IEEE Congress on Evolutionary Computation, pp. 1945–1950, 1999.

Solis, F., Wets., R., 'Minimization by Random Search Techniques', Mathematics of Operations Research, 6:19–30, 1981.

Chiping, T., McKinley, P.K., 'Energy Optimization under Informed Mobility', Parallel and Distributed Systems, IEEE Transactions on , vol.17, no.9, pp.947–962, Sept. 2006

Tu, Z., Wang, Q., Qi, H., Shen, Y., 'Flocking based distributed self-deployment algorithms in mobile sensor networks', Journal of Parallel and Distributed Computing, 72(3):437–449, March 2012.

Wan, P.J., Yi, C.W., 'Coverage by randomly deployed wireless sensor networks', IEEE Transactions on Information Theory, 52(6):265–2669, June 2006.

Wang, B., 'Coverage Control in Sensor Networks', Computer Communications and Networks, Ed.1, 1617–7975, Springer Verlag, 2010.

Wang, G., Cao, G., La Porta T., Zhang, W., 'Sensor relocation in mobile sensor networks', INFOCOM 2005, vol.4, pp.2302–2312, 2005.

Wang, G., Cao, G., La Porta, T. F., 'Movement–Assisted Sensor Deployment', IEEE Transactions on Mobile Computing, 5(6):640–652, June 2006.

Wang, L., Xu, Q. 'GPS-Free Localization Algorithm for Wireless Sensor Networks', Sensors 2010, 10, 5899–5926

Wang, X., Ma, J., Wang, S., Bi, D., 'Distributed Energy Optimization for Target Tracking in Wireless Sensor Networks', IEEE Transactions on Mobile Computing, 9(1):73–86, Jan. 2010.

Wang, X., Wang, S., Junjie, M., 'Dynamic Deployment Optimization in Wireless Sensor Networks', Intelligent Control and Automation, LNCIS 344, pp. 182–187, 2006.

Wang, X., Wang, S., Ma, J.-J., 'An Improved Co-evolutionary Particle Swarm Optimization for Wireless Sensor Networks with Dynamic Deployment', Sensors, 2007, 7(3):354–370.

Wang, Y.-C., Hu, C.-C., Tseng, Y.-C., 'Efficient Placement and Dispatch of Sensors in a Wireless Sensor Network', IEEE Transactions on Mobile Computing, 7(2):262–274, Feb. 2008.

Williams, R., 'The geometrical foundation of natural structure', Dover Publications, 1979.

Wu, H., Wang, C., Tzeng, N.F., 'Novel self-configurable positioning technique for multi-hop wireless networks', IEEE/ACM T. Network 2005, 13, 609–621.

Wu, P., Gao, L., Zou, D., Li, S. 'An improved particle swarm optimization algorithm for reliability problems', ISA Transactions, 50(1):71–81, Jan. 2011.

Xiao, Y., Chen, H., KuiWu, W., Sun, B., Liu, C., 'Modeling Detection Metrics in Randomized Scheduling Algorithm in Wireless Sensor Networks', IEEE Wireless Communications and Networking Conference, pp. 3741–3745, March 2007.

Xing, G., Wang, X., Zhang, Y., Lu, C., Pless, R., Gill, C., 'Integrated coverage and connectivity configuration for energy conservation in sensor networks', ACM Trans. Sensor Networks, 1(1):36-72, August 2005.

Xu, K., Takahara, G., Hassanein, H., 'On the robustness of grid-based deployment in wireless sensor networks', ACM Conference on Wireless communications and mobile computing (IWCMC '06), pp. 1183–1188, 2006.

Yang, S., Dai, F., Cardei, M., Wu, J., Patterson, F., 'On connected multiple point coverage in wireless sensor networks', International Journal of Wireless Information Networks, 13(4):289–301, 2006.

Yongguo, M., Yung-Hsiang, L., Hu, Y.C., Lee, C.S.G., 'A case study of mobile robot's energy consumption and conservation techniques', Advanced Robotics. ICAR '05. Proceedings., 12th International Conference on, pp.492–497, 2005.

Younis, O., Fahmy, S., 'HEED: A Hybrid, Energy-Efficient, Distributed Clustering Approach for Ad Hoc Sensor Networks', IEEE Transactions on Mobile Computing, 3(4): 366–379, Oct. 2004.

Yu, K., Montillet, J.P., Rabbachin, A., Cheong, P., Oppermann, I. UWB location and tracking for wireless embedded networks. Signal Process. 2006, 86, 2153–2171.

Zervas, E., Mpimpoudis, A., Anagnostopoulos, C., Sekkas, O., Hadjiefthymiades, S., 'Multisensor Data Fusion for Fire Detection', Information Fusion Journal, Elsevier, 12(3):150–159, July 2011.

Zhan, Z.-H., Zhang, J., Li, Y., Chung, S.-H., 'Adaptive Particle Swarm Optimization', IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics, 39(6):1362–1381, 2009.

Zhang, H., Hou, J. 1Maintaining Sensing Coverage and Connectivity in Large Sensor Networks', Ad Hoc & Sensor Wireless Networks, 1(1–2), 2005.

Zou, Y., Chakrabarty, K., 'A distributed coverage- and connectivity- centric technique for selecting active nodes in wireless sensor networks', IEEE Transactions on Computers, 54(8): 978–991, 2005.

Zou, Y., Chakrabarty, K., 'Sensor deployment and target localization in distributed sensor networks', ACM Transactions on Embedded Computing Systems, 3(1): 61–91, Feb. 2004.

Zou, Y., Chakrabarty, K., 'Sensor deployment and target localization based on virtual forces', In Proc. IEEE INFOCOM'03, Mar.-Apr. 2003, pp. 1293–1030.

Appendix

ALGORITHM 4: The ABC Algorithm

Input:
population size m (number of employed bees),
colony size $M = 2m$ (employed bees plus onlooker bees),
dimension $D = 2n$, maximum number of iterations T.
Output: optimal solution $\mathbf{p_g}$
Begin
  $\mathcal{X} = \emptyset$ {the solutions set}
  for $i = 1$ **to** m **do**
    Random $\mathbf{p}_i \in \mathbb{R}^D$
    $\mathcal{X} \leftarrow \mathcal{X} \cup \{\mathbf{p}_i\}$
  end for
  $t = 1$ {iteration index}
  repeat
    for all $i = 1$ **to** m **do**
      $\psi \sim U(-1, 1)$ {a uniformly random number in [-1,1]}
      Random $\mathbf{q}_i \in \mathbb{R}^D$
      $\mathbf{u}_i \leftarrow \mathbf{p}_i + \psi\,(\mathbf{p}_i - \mathbf{q}_i)$;
      $\mathbf{p}_i \leftarrow \arg\max(J(\mathbf{u}_i), J(\mathbf{p}_i))$
      {the ith employed bee stores its current best solution}
    end for
    $\mathcal{P} = \emptyset$ {the probabilities set}
    for $i = 1$ **to** m **do**
      $P_i = \frac{J(\mathbf{p}_i)}{\sum_{j=1}^{m} J(\mathbf{p}_j)}$;
      $\mathcal{P} \leftarrow \mathcal{P} \cup \{P_i\}$
    end for
    for all $i = 1$ **to** m **do**
      $\mathbf{p}_i = \text{selectSource}(\mathcal{X}, \mathcal{P})$
      {select randomly a food source from $\mathcal{X}$ w.r.t. $\mathcal{P}$}
      $\psi \sim U(-1, 1)$ {a uniformly random number in [-1,1]}
      Random $\mathbf{q}_i \in \mathbb{R}^D$
      $\mathbf{u}_i \leftarrow \mathbf{p}_i + \psi\,(\mathbf{p}_i - \mathbf{q}_i)$;
      $\mathbf{p}_i \leftarrow \arg\max(J(\mathbf{u}_i), J(\mathbf{p}_i))$
      {the ith onlooker bee stores its current best solution}
    end for
    for all $\mathbf{p} \in \mathcal{X}$ **do**
      if $\mathbf{p}$ is abandoned **then**
        Random $\mathbf{p} \in \mathbb{R}^D$
        $\mathcal{X} \leftarrow \mathcal{X} \cup \{\mathbf{p}\}$
      end if
    end for
    $\mathbf{p_g} \leftarrow \arg\max_{\mathbf{p} \in \mathcal{X}} J(\mathbf{p})$;
    $t \leftarrow t + 1$
  until $t > T$
End