Journal of Network and Systems Management (2024) 32:7
https://doi.org/10.1007/s10922-023-09780-x

A Novel Light Reflection-Random Walk for Smart Sensors Relocation

Nadia Belguerche[1] · Samir Brahim Belhaouari[2] · Noureddine Lasla[3] · Mahfoud Benchaïba[1]

Received: 12 May 2023 / Revised: 27 August 2023 / Accepted: 3 October 2023 /
Published online: 8 November 2023

Abstract

This paper presents a new algorithm for relocating sensors in a Wireless Sensor and Robots Network (WSRN) using a mobile robot. The goal is to repair coverage holes using redundant sensors that are caused by an initial random deployment. The holes are repaired without prior knowledge of their positions or that of the redundant sensors. The existing solutions mainly focus either on how to optimally repair holes by determining to where relocate redundant sensors, or how to build a repair path with assumption that the positions of holes and redundant sensors are known. In both scenarios, the literature lacked the optimization of the robot's path for its initial exploration to identify both the holes and redundant sensors. Our proposed solution introduces an efficient robot trajectory that utilizes stochastic paths that adhere to the principles of light reflection. This trajectory serves the dual purpose of identifying redundant sensors and detecting as well as repairing coverage holes. We achieve this by incorporating the law of large numbers into the light reflection principal, enabling the robot to move randomly while adhering to the pathways of light reflection to efficiently relocate the redundant sensors. This approach results in a highly efficient and effective sensor relocation process. The effectiveness of the proposed solution is assessed across multiple parameters, including relocation time, the length of the relocation path, the robot average moves, and the total energy consumption required to cover holes with varying carrying capacity, dimensions of region of interest, coverage ratio and exit angles of reflection. Through a series of extensive simulations, we provide compelling evidence that our proposed solution distinctly surpasses the existing state-of-the-art approaches. This notable advantage becomes evident in multiple aspects: from reduced relocation time and shorter relocation path length to minimized total energy consumption. These combined enhancements underscore the effectiveness of our solution in tackling the challenge of sensor relocation.

Keywords Robot · Coverage · WSN · Relocation · Reflection · Random walk

Extended author information available on the last page of the article

1 Introduction

Wireless sensor networks (WSNs) consist of a set of nodes with sensing, computation, and wireless communication capabilities [1, 2] deployed over a large geographical area, also known as a region of interest (ROI), to monitor physical or environmental conditions. These sensor nodes monitor and control a wide range of ambient conditions such as temperature, humidity, pressure, light and vehicular movement [3]. To meet the application requirements, sensor nodes need to be efficiently deployed to ensure continuous and optimal coverage of the entire ROI throughout the whole network lifetime.

Roughly speaking, coverage refers to the ability of sensor nodes to detect events within the monitored region. Therefore, the basic objective of nodes placement is to ensure that every point within the ROI is within the sensing range of at least one node. In this type of network, efficient network deployment can significantly improve the quality of collected data while minimizing deployment costs. Depending on the type, size, and application domain of the ROI that has to be monitored [4], the deployment of a sensor network might be deterministic (planned) or random [5]. Deterministic deployment is planned to achieve maximum coverage using the optimal number of nodes. However, to meet its goal, prior knowledge of ROI is required, which is not always available. Random deployment, however, scatters sensor nodes by aircraft and result in a randomized distribution. This kind of deployment, even if it is not accurate and incurs to deploy more sensors than necessary, it is more appropriate in harsh or hostile environments such as forests or deserts. Random deployment cannot ensure efficient coverage of the monitored ROI, where some regions can be left uncovered (coverage holes) or covered simultaneously by multiple sensors (redundant sensors). Consequently, a post-deployment that leverages the information about the current network coverage state from the already deployed sensors, is needed to complete the initial random deployment, and correct the existing coverage holes.

In the literature, two main approaches exist for recovering the initial random deployment. The first one consists in using mobile sensors that can autonomously adjust their locations to fill emerging coverage holes. However, equipping every sensor node with locomotion is expensive given the large-scale nature of the network. In addition, a mobile sensor consumes a large amount of energy due to movement which negatively affects the network lifetime. The second alternative cost-effective approach is to use a small group of mobile robots to carry the redundant static sensors and relocate them to the uncovered area (coverage holes). This problem is known as *carrier-based sensors relocation* in wireless sensors and robots networks.

In the carrier-based sensors relocation problem, robots move within the ROI to discover both sensing holes and redundant sensors and then relocate the discovered redundant sensors to the encountered holes positions. The goal is to find the optimal algorithm that ensures maximum coverage while minimizing the total path traveled by the robot. The existing work to deal with this problem

can be classified into centralized and decentralized solutions. Centralized approach requires three steps: (i) exploring and collecting all redundant sensors and holes positions from the deployed sensors and transmitting them to the central server, (ii) calculating the best robot path to repair the holes using the collected information during the first step, and (iii) repairing the holes by the robot following the path calculated in the second step. However, for ROI with large sizes, the robot requires a longer time to complete the exploration and collecting step. Consequently, the complexity of the algorithm increases with the expansion of the ROI. In the decentralized approach, however, the robot moves within the ROI to explore and at the same time repair sensing holes. On each moving step, the robot tries to discover and repair holes within its communication range and then decides its next moving step. Most existing decentralized algorithms to solve the carried-based sensor deployment problem use the least recently visited (LRV) algorithm [6]. LRV allows the robot to explore and repair holes in the ROI based only on local criteria. In LRV, the robot communicates with its neighboring sensors, which in turn provide directional guidance to the robot to place a sensor and cover holes. Usually, LRV requires many unnecessary (redundant) movements to achieve full coverage, which leads to high communication overhead and increases the holes repair time. In this paper, we opt to design a more efficient decentralized algorithm to solve the carrier-based sensor relocation problem that ensures full coverage while minimizing the robot displacement.

Our approach is novel, efficient and distinct from prior research endeavors. We incorporate both the principle of light reflection and the law of large numbers to outline the robot's movement strategy within the ROI. Light reflection is well known for its efficiency [7], precision [8] and adaptability [9]. Law of large numbers [10] allows robots to walk randomly while tracking the path of light reflection. Furthermore, our approach doesn't necessitate any preemptive knowledge regarding the positions of coverage holes or redundant sensors, which makes solution more flexible for repairing holes. In the baseline solution, the robot operates autonomously and without synchronization. It follows a light reflection path to explore and repair the ROI. The second algorithm upgrades the first one by defining a new path based on light reflection and the large number law. For this solution, the robot walks randomly while still following the light reflection path. This enables the robot to explore not only the reflection axes but also their surroundings in a single pass. This approach leads to more efficient redundant sensor collection and hole repair. This paper also assesses the performance of both algorithms by evaluating the total time needed by the robot to explore and cover holes. Our experiments have been conducted by considering ROI with large size. The rest of the paper is organized as follows. Section 2 reviews related literature work. The system model as well as the description of the concept and approach of the proposed carrier-based sensor deployment algorithms are given in Sects. 3 and 4, respectively. In Sect. 5, we present the outcomes of our simulation, and finally we conclude the paper in Sect. 6.

2 Related Works

Establishing an effective sensor deployment strategy is a core concern in the realization of a WSN designed to monitor a specific ROI and transmit the gathered data to a central base station. The deployment model determines the sensor types, number and positions necessary to construct an effective system that can fulfill the requirements of a given application scenario. The deployment strategies are mainly classified into two categories [11–13]: deterministic and random. The choice of deployment strategy is influenced by factors such as the size of the monitored area and the specific application domain, all aimed at achieving optimal coverage [4]. In the majority of cases, random deployment is the favored approach due to its simplicity and adaptability to diverse areas of interest (e.g., war zones, inaccessible regions, etc.). The latter is subdivided into two distinct types of algorithms: The first involves self-deployment algorithms, where sensors independently relocate to insure full coverage after an initial random deployment. The second type pertains to carrier-based sensor deployment, wherein mobile robots take in charge the relocation of redundant sensors that were originally placed randomly. This paper specifically focuses on the carrier-based approach. Hereafter, we present a concise review of related works concerning sensor deployment in WSNs.

Li et al. [6] introduced a family of localized algorithms: R3S2, G-R3S2 and CR3S2. In thier approach, sensors and robots are scattered randomly across the ROI. In the R3S2 algorithm, the robot moves randomly, and establishes communication with sensors within its range to identify coverage holes and redundant sensors. If the carrying capacity of the robot is zero, it focuses on locating redundant sensor; otherwise, it seeks out holes in need of coverage. In the G-R3S2 algorithm, the area is divided into a virtual grid. Akin to [14], the robot navigates between the grids, selecting the direction that has been least recently visited (LRV). In the CR3S2 algorithm, the authors adopt a Connected Dominating Set (CDS) strategy. The cluster head is determined as the sensor with the highest ID. This method involves four adjacent grids and combines the information about holes and redundant sensors from cluster members into a single information that is disseminated to all members within the cluster. While using the LRV technique to explore the ROI, these algorithms, however, necessitate numerous unnecessary movements to achieve complete coverage. Additionally, the robot's carrying capacity is constrained to one, a limitation that might not reflect real-world scenarios.

In [15, 16], the authors focused on optimizing the number of redundant sensors needed to repair a hole. The robot's exploration of the ROI relies on the LRV technique. However, the use of LRV introduces unnecessary movements which subsequently increase the relocation time. Additionally, similar to [6], the robot's carrying capacity is limited to one.

Desjardins et al. [17–19] proposed a multi-objective methodology to address the redeployment problem. In [17], they assigned a representation of 1 to each redundant sensor and −1 to the holes. The objective is to determine a replacement

trajectory for sensors using scalable multi-objective optimization (EMOO) algorithm. This algorithm takes into account the quality of the carrying passive sensors (battery capacity) or their initial positioning prior to relocation. In addition, authors use additional decision objectives such as ensuring the robot always deploys the latest-worn sensor upon reaching a coverage hole. The algorithm uses a greedy heuristic to shorten the trajectory. In [18], the previous algorithm presented in [17] was enhanced. The authors employed the Risk Management Framework (RMF) [20] for proactive sensor replacement. This methodology proactively detects high-risk sensors failure and replaces them to avert the emergence of coverage holes, thereby minimizing any potential deficiencies in coverage across the ROI. In [19], authors extend the two previous solutions [17, 18] by considering the scenario involving multiple robots. An additional objective function is introduced to ensure load balancing among participating robots. However, the complexity increases with higher number of sensor to be replaced. Consequently, these algorithms may not be suitable in contexts involving extensive sensor relocation requirements.

In [21], the approach involves partitioning the ROI into grids. The robot first passes through the entire ROI to collect the number of redundant sensors and the number of coverage holes within each grid. Based on these metrics, the robot path is build to cover holes in the ROI. To repair the holes, redundant sensors from adjacent grids are utilized. If the number of redundant sensors in the adjacent grids is not enough, the solution extends to secondary level grids and so on, until a sufficient quantity of redundant sensors is obtained for hole reparation within a grid. However, in this solution, although the robot possesses a global view of the network,it does not fully exploit it. Instead, it resorts to a local view for repair, which might not consistently yield the most optimal path.

In [22], theissue is approached as an extension of the traveling salesman problem (TSP), aiming at achieving a more efficient solution for the relocation problem in a large-scale scenario. The authors use Lin–Kernighan–Helsgaun (LKH) heuristic to form a Hamiltonian cycle. The robot path is thus constructed with a constraint that the robot only passes through a hole if it carries a redundant sensor, and respectively it only passes through a redundant sensor if its carrying capacity has not been reached. Nonetheless, it's important to note that the LKH heuristic operates through iterations. Consequently, as the path length increases, so does the complexity of the process.

In [23], the robot replaces damaged sensors by picking up spare ones from the region of interest or carrying them from a central station. By using the LKH algorithm, the shortest Hamiltonian cycle of all delivery nodes (holes) is first constructed. The baseline path is then modified by adding passive sensors while minimizing its length.

3 Background

In this section, the basic techniques employed in the proposed algorithms are briefly described. These techniques include the principle of light reflection [24, 25] and the law of large numbers [10].

3.1 Reflection of Light

When light bounces a smooth, shiny surface, it is reflected at the same angle at which it struck the surface. The ray that undergoes this reflection is called the reflected ray, the point where the surface is hit is called the point of incidence, and the line perpendicular to this point is called the normal. The angle formed by the ray of incidence with the normal is called the angle of reflection. Figure 1 shows an illustrative example of light reflection. This principal of light reflection is applied within our algorithm to enable the robot to efficiently navigate and explore the ROI.

3.2 Law of Large Numbers

In statistics and probability theory, the law of large numbers [10, 26] stipulates that when a given experiment is repeated a large number of times, the average of the results should be close to the expected value. Additionally, in statistics, the central limit theorem states that if you sufficiently repeat an experiment using random samples, the mean of these samples tends to follow a normal distribution. As a result, the samples' mean will be closer to the central line, or the line through which the mean of a normal distribution curve is marked [27]. It is demonstrated in practice that this theorem is true whenever the number of experiments, n, is greater or equal to 30 [28].

In our case, the objective is to explore the ROI based on the principle of random walk with a reflection light trend. The robot repeats the same experiment to explore the ROI a large number of times given the size of the ROI to be monitored. So, the average of the results of this walk should be close to the reflection of light principle.

4 System Model

We consider a WSN, where static sensor nodes and a mobile robot are randomly scattered throughout the ROI. Let $S = \{S_0, S_1, \ldots, S_{n-1}\}$ be the set of n static sensors. A sensing hole is an area that is continuously uncovered. A sensor whose sensing range is fully overlapped by other neighbor sensors is defined as a redundant sensor. We assume that there are sufficient redundant sensors to repair all ROI holes.

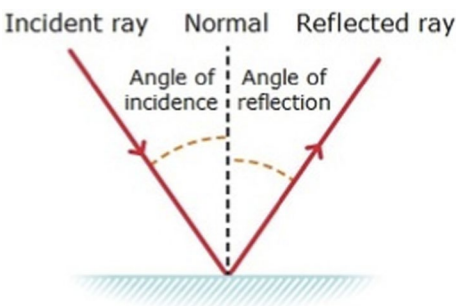


Fig. 1 Reflection of light

The proposed algorithms allow the robot to carry redundant sensors to cover holes. A robot's carrying capacity is limited to the number of static sensors it can carry. For simplicity, similar to that in [6, 11–13], we assume that nodes (sensors and robot) have circular-shaped coverage. Identical sensors have the same sensing circle rs and communication circle rc, where $rc \geq 2rs$ ensures all sensors with overlap are connected. The robot has a communication range Rc, where $Rc > rc$. Direct communication can take place between any two nodes, only when they are within each other's transmission range. We also assume that the position of every robot and sensor is known by using a localization system, such as GPS. For the sake of simplicity we model the ROI as a square of side length $M = L$ (where M and L are the lengths of the area in the xx and yy directions, respectively).

5 Efficient Relocation Algorithms

In this paper, we propose two algorithms that deal with the carrier-based sensor deployment problem. The algorithms allow the robot to collect redundant sensors and place them in uncovered areas. In the first variant of our algorithm: Reflection based sensor relocation algorithm (R-SR), the robot explores the area using the reflection of light principle to localise and repair the coverage holes in the ROI with localised redundant sensors. This principle ensures that the robot visits the entire ROI. This allows it to discover and repair all uncovered areas in the ROI, ensuring full coverage. In the second variant: Light Reflection based sensor relocation algorithm (LR-SR), instead of the robot exploring only the axis of reflection, it makes a random walk around the axis generated by the law of reflection to explore the surroundings of the axis. As a result of this exploration, the robot can repair more holes (using redundant carried sensors) within a single passage, reducing relocation time.

Sensors and a robot are randomly scattered throughout the ROI. Through periodic 'hello' messages, each sensor identifies if it is redundant or hole boundary. In the literature there are several algorithms which deal with holes detection and redundant sensors detection which can be directly applied, so we have not addressed this point in our paper. For example, the algorithm proposed by [29–31] can be used to allow a sensor to detect if it is a redundant sensor and [32–34] to detect if it is a hole boundary.

5.1 R-SR Algorithm

In R-SR, the robot begins a research process from a given point of the ROI, which consists of moving step by step to localize the existing redundant sensors and holes. The robot didn't need any initial information about the ROI or sensors, it moves autonomously and asynchronously.

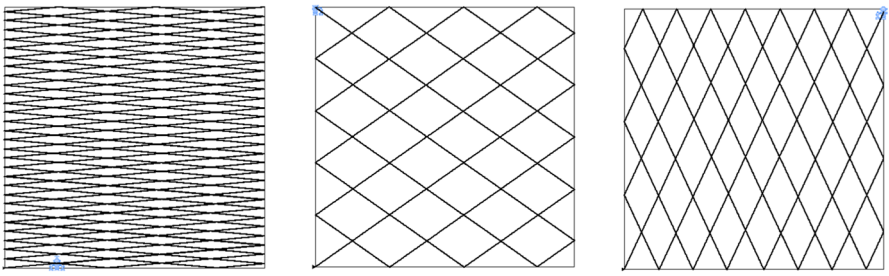
The robot moves for distance Δ along the light reflection axes. Then, it stops moving and communicates with sensors in its communication range to collect information about redundant sensors and holes in their neighboring. Basing on this information, the robot repairs as possible (if there is enough redundant sensors) the

detected holes using the carrying redundant sensors and the detected redundant sensors. Once the robot repairs holes, it moves for another Δ along the reflection axes and repeat this process until the full coverage is reached.

Initially, the robot starts exploring at a point on the ROI's border with an exit angle of α. The choice of the angle α and start point are very critical and need to be optimized. If the right exit angle is chosen, few light reflection axes are sufficient for the robot to cover all the ROI. Otherwise, it needs more light reflection axes to cover the ROI. The right angle depends on the robot's communication radius, since the angle defines the distance between light reflection axes. If the distance between light reflection axes is more than 2*RC, the robot is uninformed of all holes and redundant sensors. The sensors located at a distance greater than RC from the first reflection axe and the second reflection axe are unable to inform the robot of the presence of holes and redundant sensors in their neighbors. Therefore, the full ROI coverage is not guaranteed. On the other hand, if the distance between reflections is small, the relocation process is longer. The robot stops several times to collect information already collected. Figure 2 shows an example of the variation of the exit angle: Fig. 2a shows the light reflection axes with exit angle = 5°, Fig. 2b shows the light reflection axes with exit angle = 35° and Fig. 2c shows the light reflection axes with exit angle = 65°. After selecting the appropriate exit angle and point of start, the robot follows the trajectory of the axis generated by the law of light reflection. After each distance Δ on the axis, the robot communicates with its neighbors to verify the existence of holes and redundant sensors. In accordance with its carrying capacity, it determines whether it can repair the holes and if it can collect the redundant sensors. If the robot carrying capacity is less than C, it carries the detected redundant sensors without exceeding C. As long as the robot is carried with redundant sensors, it repairs detected holes. The robot repeats this process after each step Δ. When the robot reaches a border, it changes direction with the *alpha* angle as shown in Fig. 2. It starts with a new axis.

5.2 Algorithm LR-SR

In this algorithm, the law of large numbers [10, 26] was introduced to permit the robot to explore the ROI randomly while trending according to the light reflection



(a) $Angle = 5°$ (b) $Angle = 35°$ (c) $Angle = 65°$

Fig. 2 Three reflection scenarios with different angles

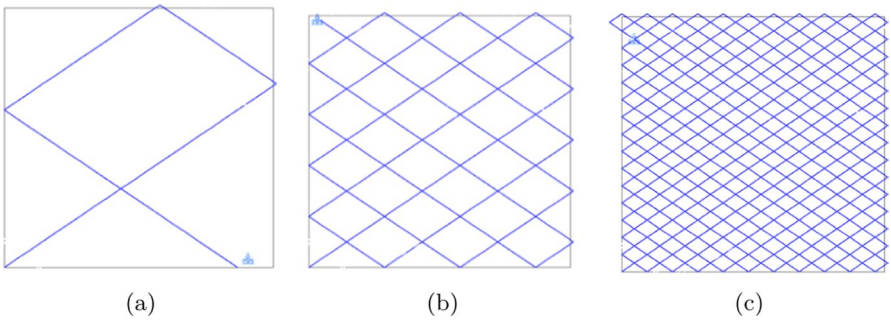


Fig. 3 Reflections' evolution over time

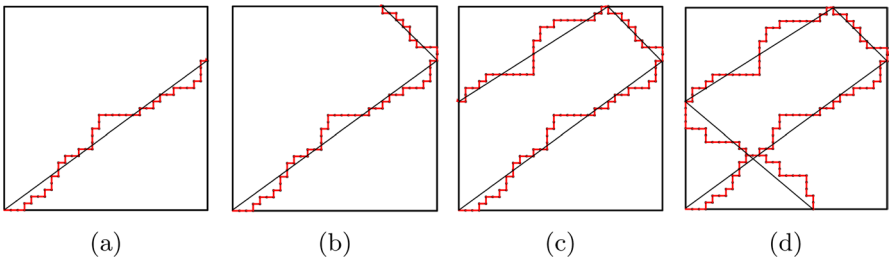


Fig. 4 Illustrative example of the robot's random walk

law. In contrast to the R-SR algorithm, where the robot simply explores the reflection axis, the robot in the LR-SR algorithm explores both the light reflection axis and its surroundings. Figure 4 shows the exploration of ROI with LR-SR algorithm. The goal of exploring around the axis of reflection is to decrease the number of reflection axis needed to explore and repair all holes. This will decrease the time needed to relocate redundant sensors to cover all holes.

In light reflection, the reflection axes get closer as the robot explores the ROI. The reflection axes begin quite far apart, but as the robot explores the area and creates more reflection axes, they grow closer together. This is shown in Fig. 3. The LR-SR algorithm does not require the reflections to be too close because the robot explores the reflection axis and its surroundings, in contrast to the R-SR algorithm where the robot explores only the reflection axis. As a result, the LR-SR algorithm is less need for reflection axes during the repair process. As the robot explores the ROI step by step, it communicates with its neighbors to collect redundant sensors and repairs holes. After the robot repairs the detected holes, it moves to the next step. At each step, the robot chooses between two directions. For example, if the robot explores an axis which goes from the left border to the lower border, it decides between walking towards the south or the east from its position. The robot selects a probability at random, compares it to two probabilities P_1 and P_2 that match with the two directions. It then chooses the direction with the closest probability to the random probability. The probabilities P_1 and P_2 are calculated so that the robot's path trends the light reflection axes trajectory. Given the vast and large

size of the ROI, the robot stops several times to choose a probability p and then selects an exploration direction. As explained in Sect. 3, when the same experiment is repeated a large number of times, the average of the test results should be close to the expected result, which is appropriate with this kind of the ROI. The more steps, the more closely the robot path's matches the light reflection axes trajectory as shown by Fig. 4.

The robot is moving according to a path of a random walk in two dimensions where its position at time n in noted as $\begin{pmatrix} X_n \\ Y_n \end{pmatrix}$. Each coordinate of the robot can be seen as random walk in one dimension defined as follows:

$$\begin{cases} X_n = \sum_{i=1}^{n} x_i \\ Y_n = \sum_{i=1}^{n} y_i \end{cases}$$

where x_i and y_i are independent random variable for integers i, and both are identically distributed random variables following one of the following probability mass functions:

$$P(x_i = x) = \begin{cases} P_1 \ if \ x = \Delta \\ 1 - P_1 \ if \ x = 0 \end{cases} and \ P(y_i = y) = \begin{cases} 1 - P_2 \ if \ y = 0 \\ P_2 \ if \ y = \Delta \end{cases}$$

Or

$$P(x_i = x) = \begin{cases} P_1 \ if \ x = -\Delta \\ 1 - P_1 \ if \ x = 0 \end{cases} and \ \ P(y_i = y) = \begin{cases} 1 - P_2 \ if \ y = 0 \\ P_2 \ if \ y = \Delta \end{cases}$$

Or

$$P(x_i = x) = \begin{cases} P_1 \ if \ x = \Delta \\ 1 - P_1 \ if \ x = 0 \end{cases} and \ P(y_i = y) = \begin{cases} 1 - P_2 \ if \ y = 0 \\ P_2 \ if \ y = -\Delta \end{cases}$$

Or

$$P(x_i = x) = \begin{cases} P_1 \ if \ x = -\Delta \\ 1 - P_1 \ if \ x = 0 \end{cases} and \ P(y_i = y) = \begin{cases} 1 - P_2 \ \text{if} \ y = 0 \\ P_2 \ if \ y = -\Delta \end{cases}$$

At case of the above, the robot is following a linear trend is his moving with an random zigzag.

Let θ is the incident angle and the initial exit angle of the robot from the initial point, y_i and x_i are identical independent distributed random variables as follow:

$$\begin{pmatrix} X_n \\ Y_n \end{pmatrix} = \sum_{i=1}^{n} \overrightarrow{X_i} = \sum_{i=1}^{n} \begin{pmatrix} x_i \\ y_i \end{pmatrix}$$

$$\tan(\theta) = \lim_{n \to \infty} \frac{\sum_{i=1}^{n} y_i}{\sum_{i=1}^{n} x_i} = \lim_{n \to \infty} \frac{Y_n}{X_n} = \frac{P_2}{P_1},$$

where X_n and Y_n are the width and the length of the ROI. On the other hand : $P_1 + P_2 = 1$, then the two equations were deduced:

$$\tan(\theta) = \frac{P_2}{P_1}, \tag{1}$$

$$P_1 + P_2 = 1. \tag{2}$$

From (1) and (2) we deduce that:

$$P_1 = \frac{1}{1 + \tan(\theta)},$$
$$P_2 = \frac{\tan(\theta)}{1 + \tan(\theta)}.$$

Once the probabilities P_1 and P_2 are calculated, the robot moves step by step. In each step, it randomly chooses a probability p in the interval [0, 1]. This probability p is compared with P_1 and P_2 which allows the robot to choose a direction forward to explore the ROI with the principle of reflection using a random walk. The robot moves with distance Δ on one of the four directions (north, south, east or west) according to the probability p and its previous step. Where Δ is any positive real number chosen in terms of ROI size (Length M, wide L), $\Delta < max(L, M)/100$. The possible cases and rules to decide on the next step are as follows:

1. If the last movement of the robot was from south to north or from left to right (Fig. 4a) and $p \leq P_1$, then the next step is to move with Δ to right. Else, if $p > P_1$, then the next step is to move to north with Δ as shown by function 3.

$$\vec{X}_{i+1} = \begin{cases} \vec{X}_i + \begin{pmatrix} \Delta \\ 0 \end{pmatrix} \text{ with } P_1 \\ \vec{X}_i + \begin{pmatrix} 0 \\ \Delta \end{pmatrix} \text{ with } P_2 \end{cases} \tag{3}$$

2. If the last movement of the robot was from south to north or from right to left (Fig. 4b) and $p \leq P_1$ then the next step is to move to north. Else, if $p > P_1$ then the next step is to move to left as shown by Eq. (4).

$$\vec{X}_{i+1} = \begin{cases} \vec{X}_i + \begin{pmatrix} -\Delta \\ 0 \end{pmatrix} \text{ with } P_1 \\ \vec{X}_i + \begin{pmatrix} 0 \\ \Delta \end{pmatrix} \text{ with } P_2 \end{cases} \tag{4}$$

3. If the last movement of the robot was from north to south or from right to left (Fig. 4c) and $p \leq P_1$ then the next step is to move to south. Else, if $p > P_1$ then the next step is to move to left with Δ as shown by Eq. (5).

$$\vec{X}_{i+1} = \begin{cases} \vec{X}_i + \begin{pmatrix} -\Delta \\ 0 \end{pmatrix} \text{ with } P_1 \\ \vec{X}_i + \begin{pmatrix} 0 \\ -\Delta \end{pmatrix} \text{ with } P_2 \end{cases} \tag{5}$$

4. If the last movement of the robot was from north to south or from left to right (Fig. 4d) and $p \leq P_1$ then the next step is to move to south. Else, if $p > P_1$ then the next step is to move to right with Δ as shown by Eq. (6).

$$\vec{X}_{i+1} = \begin{cases} \vec{X}_i + \begin{pmatrix} \Delta \\ 0 \end{pmatrix} \text{ with } P_1 \\ \vec{X}_i + \begin{pmatrix} 0 \\ -\Delta \end{pmatrix} \text{ with } P_2 \end{cases} \tag{6}$$

This is repeated for each step until the robot reaches one of the borders of the ROI. At the border, the robot randomly chooses a new probability p within the interval [0, 1] and compares it to the probabilities P_1 and P_2. According to the robot's coordinates (x, y) at the border, it decides its next movement direction as follow:

1. If $x = M$, where M is the width of the ROI (i.e., it reaches the right border of the ROI), then it applies the rule in Eq. (3).
2. If $y = L$, where L is the length of the ROI, then it applies the rule in Eq. (4).
3. If $x = 0$, then it applies Eq. (5).
4. If $y = 0$, then Eq. (6) is applied.

The algorithm terminates when the ROI is fully covered.

Figure 5a shows the ROI initial state after a random deployment of sensors and the robot. The example of ROI repair using the R-SR method is shown in Fig. 5b, whereas an illustration of reparation using the LR-SR method is shown in Fig. 5c. In Fig. 5b, the robot moves on the axes generated by the reflection light (represented in the figure by blue lines). However, in Fig. 5c the robot moves randomly while its path trends the light reflection axes (the path is represented by blue lines). According to the Fig. 5b, c, the robot needs 10 reflection axes to cover the ROI when using the R-SR algorithm, however with the LR-SR method, the robot only needs six reflection axes to cover the same ROI with the same starting deployment.

θ= Exit angle
(x, y) = current position of the robot
repeat
  if $\exists$ *Holes* $\|$ *Redundant sensors in RC* **then**
    Reparation(Holes, Redundant sensors)
  end if
  $x_i = \Delta * \cos(\theta)$
  $y_i = \Delta * \sin(\theta)$
  $(x, y) = (x, y) \pm (x_i, y_i)$
  Move to (x, y)
until End of process

Algorithm 1 R-SR Algorithm

θ = Exit angle
P1 = 1 / (1 +tan(θ))
$P2 = \tan(\theta)/(1 + \tan(\theta))$
(x, y)= current position of the robot
repeat/*Repeat after each step (Δ)*/
  if $\exists$ *Holes* $\|$ *Redundant sensors in RC* **then**
    Reparation(Holes, Redundant sensors)
  end if
  $p \leftarrow Random \in [0, 1]$
  if $p \leq P_1$ **then**
    $(x, y) = (x, y) \pm (\Delta, 0)$
    Move to (x, y)
  else
    $(x, y) = (x, y) \pm (0, \Delta)$
    Move to (x, y)
  end if
until End of process

Algorithm 2 LR-SR Algorithm

5.3 Complexity Analysis

In this subsection, the complexity of the proposed algorithm is analyzed. The complexity of our algorithm can be calculated in a simple way using trigonometric rules. Let *Cr* be the complexity for one round, *Nbr* number of rounds need to cover the ROI and *Cg* the complexity for all grid. The complexity of the proposed algorithm $Cg = Cr \times Nbr$.

In the first step, the complexity for one round (Fig. 4d) was calculated. Let Δ be the scale, $\Delta \in R^+$ and the grid size of the ROI be $L \times M$. In each round, the robot jumps in one of the four directions (up, left, down or right). The complexity of one

while There are still holes and redundant sensors **do**
  if Capacity == 0 && RedundantSensors!=NULL **then**
    NextPosition ← NearestPosition(RedundantSensors)
    $Capacity \leftarrow Capacity + 1$
  else if Capacity > 0 and Capacity < CapacityMax **then**
    $NextPosition \leftarrow NearestPosition(Holes \cup RedundantSensors)$
    if NextPosition==Redundant **then**
      $Capacity \leftarrow Capacity + 1$
    else if NextPosition==Hole **then**
      $Capacity \leftarrow Capacity - 1$
    end if
  else if Capacity==CapacityMax && Holes!=NULL **then**
    $NextPosition \leftarrow NearestPosition(Holes)$
    $Capacity \leftarrow Capacity - 1$
  end if
end while

Algorithm 3 Reparation Algorithm

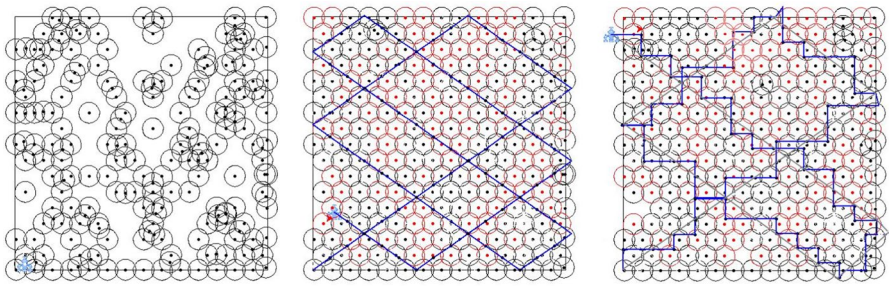


Fig. 5 Illustrative example of the two algorithms

round *Cr* is calculated by adding all jumps in one round, where the number of jumps on each direction is:

- Number of jump in ↑ : $\frac{L}{\Delta}$.
- Number of jump in ← : $\frac{M}{\Delta}$.
- Number of jump in ↓ : $\frac{L}{\Delta}$.
- Number of jump in → : $\frac{M}{\Delta}$.

$$Cr = \frac{2(L+M)}{\Delta} = \frac{perimeter\ of\ the\ grid}{\Delta}.$$

In the second step, the number of rounds needs to cover all grid Nbr was calculated as follow:

$$Nbr = \frac{L}{\cos(\theta)2\Delta}.$$

Finally, the complexity to cover the entire grid was calculated as follow:

$$Cg = O\left(\frac{2(L+M)}{\Delta}\frac{L}{\cos(\theta)2\Delta}\right) = O\left(\frac{(L+M)L}{\Delta^2\cos(\theta)}\right).$$

6 Performance Evaluation

As mentioned previously, distributed carrier-based sensor relocation algorithms were proposed, where sensors are deployed by aircraft/helicopter airdrops. The comparison of the proposed algorithms with two other algorithms from the literature was done: LRV algorithm, since this exploration technique was used by many distributed solutions to the carrier-based sensor problem [6, 14, 15]. The second algorithm CB-SD [21] which is a recent solution to the relocation problem. To do so, the performances of all these algorithms (i.e., the proposed algorithms, LRV and CB-SD) were evaluated, by varying the exit angle and the coverage ratio while fixing the ROI surface to 1080×1080, the robot's step Δ to $4 \times Rs$ the robot's carrying capacity to 8. We then varied the carrying capacity and the ROI surface while fixing the exit reflection angle to 35°. Our simulator was developed using Python. Initially, sensors are randomly deployed on a two-dimensional plane. The initial coverage ratio is equal to 65% which is increased gradually. To evaluate our algorithms, the following performance metrics were used :

- **The relocation time (Tr)**: is the time required for the relocation of the redundant sensors in the detected coverage holes to increase the initial coverage rate to achieve 99%.
- **The coverage rate (Tc)**: the percentage of the area covered by at least one sensor over the entire area of ROI.
- **Average distance traveled by the robot (Dm)**: displacement distance average of the robot during the simulation (exploration and repair).

6.1 Experiment 1: The Relocation Time Versus Delta

In the first experiment, the relocation time of the four algorithms was evaluated by varying the length of the robot's step (Δ).

As shown in Fig. 6, the larger the robot's step (Δ), the more the relocation time decreases, until Δ reaches an optimal step of $4 \times rs$. Beyond this step, relocation time increases. This is explained by the fact that in the four algorithms the robot stops after each step to collect redundant sensors and/or to repair holes (LRV, LR-SR, R-SR) or to collect information (CB-SD) in its communication radius. The amount of repeat information on redundant sensors and holes increases when the step decreases below RC. As noted, LR-SR and R-SR algorithms give better

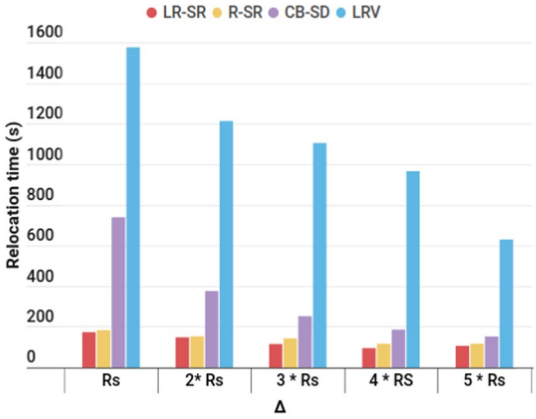


Fig. 6 Impact of variation of Δ on relocation time

results than the other two algorithms. This is explained by the fact that in CB-SR the robot must first explore the full ROI before starting repairing. In the LRV algorithm, the robot revisits the same zone several times, unlike our solutions where the principle of light reflection makes it possible to repair the ROI holes by walking randomly around a few light reflection axes with Δ equal to $4 \times Rs$. The random walk allowed the robot to reduce the relocation time by more than 19% (from 117.1 to 95.19 s) compared to the time taken by the walk following the light reflection. The introduction of random walk around the axis allows the robot to explore not only the axis but also its surroundings in a single pass. This reduces the total relocation time and confirms our choice of using a random walk around the axis generated by light reflection in our solution (LR-SR).

6.2 Experiment 2: The Relocation Time Versus the Reflection Exit Angle

In the second experiment, the relocation time of our two algorithms was evaluated by varying the reflection exit angle.

The histogram in Fig.7 shows the impact of varying exit angles on the relocation time. The angle 5° is the worst case with a relocation time equal to 156,32 s with LR-SR and 136.89 s with R-SR. While the relocation time produced by other angles varies between 120.50 and 127.45 s. The exit reflection angle 35° gives the best relocation time which is equal to 117.1 s with R-SR and 95.19 s with LR-SR. By adopting the light reflection principle, the robot divides the ROI into sub-areas. In LR-SR, the robot explores these sub-areas following a random walk around the axes generated by this principle, to detect coverage holes and redundant sensors in a radius Rc from the axis. In R-SR, the robot explores only the axes. With the angle 35°, the ROI is divided in an optimal way. The rate of overlap of one axis exploration with the next is optimal. Unlike the very small angles which bring the reflection axes closer together. Consequently, it leads to slow exploration since the robot explores almost the same subareas of the two axes. Especially in the LR-SR solution, where the robot explores the surroundings of the axis, this takes more time than the R-SR algorithm.

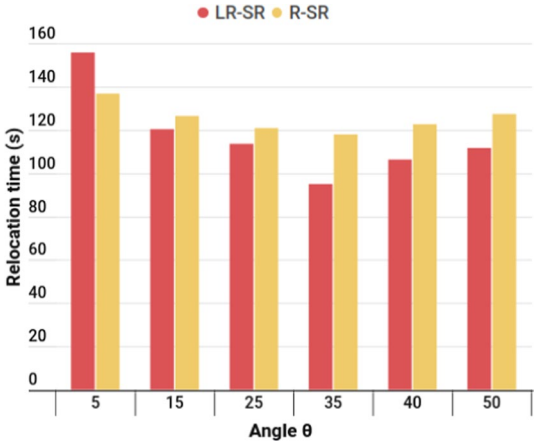


Fig. 7 Impact of variation of reflection exit angle on relocation time

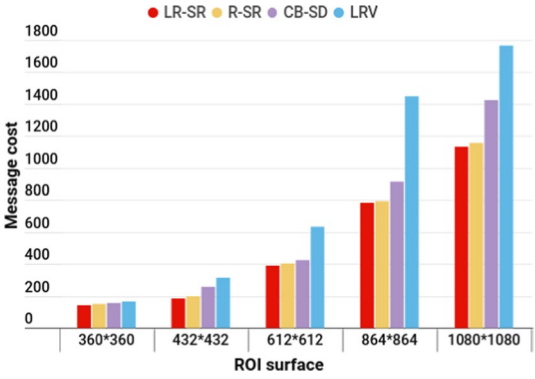


Fig. 8 Impact of variation of ROI surface on the messages cost

6.3 Experiment 3: The Relocation Time, Messages Cost and Robot Average Moves Versus the ROI Surface

In the third experiment, the relocation time and the robot average moves of the four algorithms was evaluated by varying the ROI surface.

From Figs. 8, 9 and 10, it is clear that the larger the surface, the greater the difference between the solutions. This means that LRV and CB-SD are not adapted to large areas. Even if R-SR gives better results than LRV and CB-SD, the LR-SR approach is the most effective compared to other approaches and keeps an almost linear increase in relocation time, messages cost and robot average moves. As a result of the unnecessary (redundant) movements of LRV, the robot re-explores an area already explored instead of going to new areas that have not been explored before. In the CB-SD, the robot explores the ROI first to collect the positions of redundant sensors and holes before starting the repair process. First steps in large areas take time, so relocation times, message costs, and robot average moves increase due to the increased robot average moves. The R-SR method explores the

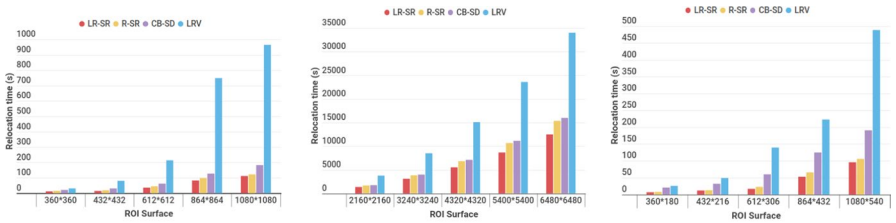
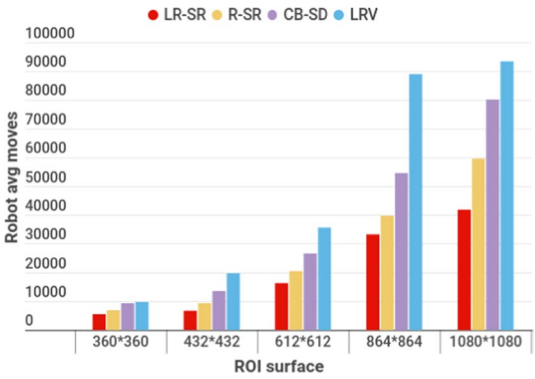


Fig. 9 Impact of variation of ROI surface on the relocation time

Fig. 10 Impact of variation of ROI surface on robot average moves



ROI and repairs holes following the light reflection axes. However, LR-SR explores and repairs holes surrounding the axes which allows the robot to repair more holes in one pass. Since random deployment of sensors was involved, this means that the application in question takes place in a large area. Our solution is better suited to this type of application.

6.4 Experiment 4: The Relocation Time Versus Coverage Ratio

In the fourth experiment, the relocation time of the four algorithms was evaluated by varying final coverage ratio.

As shown by Fig. 11, the relocation time required for our solution (LR-SR) to achieve any coverage rate is much smaller than the time required for LRV, CB-SD and R-SR to reach that same rate. As explained above, the robot in LRV is subject to search loops. In CB-SR the robot explores first the ROI to discover the positions of redundant sensors and holes and due to the size of ROI, it will take more time. Our solutions (R-SR and LR-SR), work differently. The robot repairs the holes with the redundant sensors collected during the ROI exploration. The fact that in LR-SR the robot discovers the surroundings of the reflection axis instead of discovering only the axis like in R-SD, implies that the robot in LR-SD detects more holes and

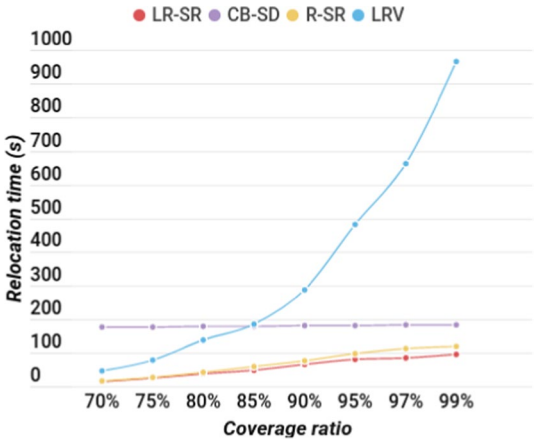


Fig. 11 Impact of variation of coverage ratio on relocation time

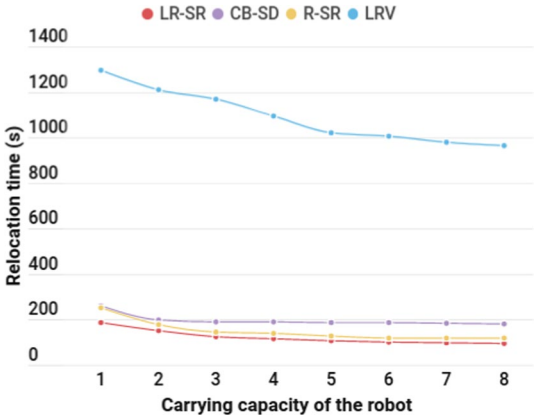


Fig. 12 Impact of variation of carrying capacity of the robot on relocation time

redundant sensors in a single pass. This explains why LR-SR gives better results than R-SR.

6.5 Experiment 5: The Relocation Time Versus Carrying Capacity of the Robot

In the fifth experiment, the relocation time of the four algorithms was evaluated by varying the carrying capacity of the robot.

As shown in Fig.12, in all four algorithms, as the robot carrier capacity increases, the relocation time decreases to reach a coverage rate equal to 99%. The increased carrying capacity of the robot allows it to carry several sensors at once, and thus to repair large coverage holes without searching for other redundant sensors. As previously stated, the robot's exploration of the ROI and reparation process enables our algorithm to provide the best time relocation when compared to other algorithms.

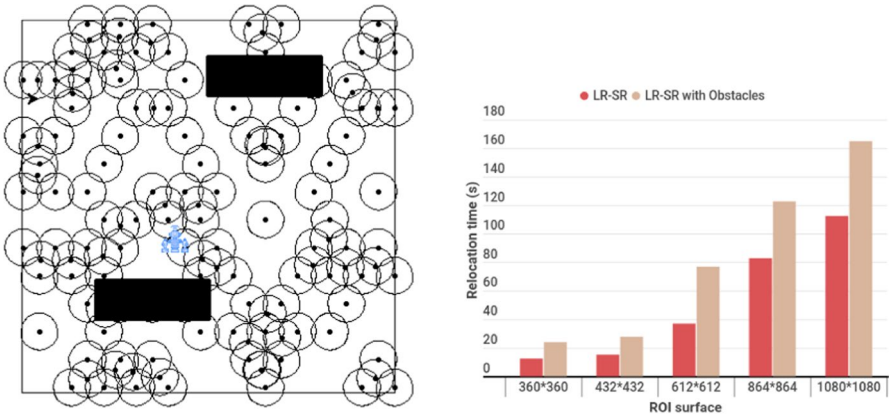


Fig. 13 Impact of the presence of obstacles on LR-SR algorithm

6.6 Experiment 6: Impact of the Presence of Obstacles on LR-SR Algorithm

The sixth experiment focuses on assessing the influence of obstacles within the ROI, as shown in Fig. 13a, on the LR-SR algorithm. The relocation time of the LR-SR algorithm in both scenarios with and without obstacles was assessed by varying the ROI's surface.

The LR-SR algorithm ensures maximal coverage even in the presence of obstacles, as demonstrated in Fig. 13b. Relocation time increases with ROI surface size regardless of the presence or absence of obstacles in the ROI. This is explained by the fact that the robot must spend more time addressing all of the holes within a region of interest as the surface area increases. Additionally, the obstacles in the ROI also increases relocation time. This is due to the fact that the robot has to re-route its path to avoid obstacles in the environment. As a result, relocation time increases.

7 Conclusion

In this paper, a localized algorithm was designed to solve the carrier-based sensors relocation problem. First a basic solution R-SR was proposed which is based on the light reflection principle. The light reflection allows the robot to have a general view of the network concerning the holes and the redundant sensors in just a few reflection axes. Therefore, the relocation process is more efficient in terms of distance traveled by the robot and relocation time. To optimize the distance and relocation time, the LR-SR algorithm was proposed. To allow the robot to repair more holes on one pass, the law of large numbers was integrated with the reflection light law. In LR-SR, the robot even explores the surroundings of the reflection axes instead

of exploring the axes only as in the R-SR algorithm. The simulation results showed that the LR-SR algorithm yields the highest performance in detecting holes in a short period of time comparing with other techniques. The LR-SR algorithm can be improved by optimizing holes reparation in each step. In future work, multi robots version will be considered.

Declarations

Conflict of interest The authors have no relevant financial or non-financial interests to disclose. The authors have no conflicts of interest to declare that are relevant to the content of this article. All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript. The authors have no financial or proprietary interests in any material discussed in this article.

References

1. Rajpoot, P., Dwivedi, P.: Optimized and load balanced clustering for wireless sensor networks to increase the lifetime of WSN using MADM approaches. Wireless Netw. **26**(1), 215–251 (2020)
2. Qabouche, H., Sahel, A., Badri, A.: Hybrid energy efficient static routing protocol for homogeneous and heterogeneous large scale WSN. Wireless Netw. **27**(1), 575–587 (2021)
3. Shaimaa, M.M., Haitham, S.H., Saroit, I.A.: Coverage in mobile wireless sensor networks (M-WSN): a survey. Comput. Commun. **110**, 133–150 (2017)
4. Debnath, S., Kumar, A., Hossain, A.: A comprehensive survey of coverage problem and efficient sensor deployment strategies in wireless sensor networks. Indian J. Sci. Technol. **9**(45), 1–6 (2016)
5. Cheng, C.F., Chen, Y.C.: The carrier-based sensor deployment in linear IWSNs with return/non-return branches. IEEE Sens. J. **22**(6), 6175–6186 (2022)
6. Li, X., Fletcher, G., Nayak, A., Stojmenović, I.: Randomized carrier-based sensor relocation in wireless sensor and robot networks. Ad Hoc Netw. **11**(7), 1951–1962 (2013)
7. Kumar, A., Srivastava, R., Kamalasanan, M.N., Mehta, D.S.: Enhancement of light extraction efficiency of organic light emitting diodes using nanostructured indium tin oxide. Opt. Lett. **37**(4), 575–577 (2012)
8. Bambi, C., Brenneman, L., Dauser, T., García, J., Grinberg, V., Ingram, A., Jiang, J., Liu, H., Lohfink, A., Marinucci, A., Mastroserio, G., Middei, R., Nampalliwar, S., Niedźwiecki, A., Steiner, J., Tripathi, A., Zdziarski, A.: Towards precision measurements of accreting black holes using X-ray reflection spectroscopy. Space Sci. Rev. **217**(5), 65 (2021)
9. Zhang, X., Zhou, Y.: Adaptive lighting algorithm experiment. In: 13th IEEE International Conference on Electronic Measurement and Instruments (ICEMI), pp. 348–353. Yangzhou, China (2017)
10. Petrov, W.W.: On the strong law of large numbers. Stat. Prob. Lett **24**(3), 1589–1615 (1996)
11. Feng, S., Shi, H., Huang, L., Shen, S., Yu, S., Peng, H., Wu, C.: Unknown hostile environment-oriented autonomous WSN deployment using a mobile robot. J. Netw. Comput. Appl. **182**, 053–103 (2021)
12. Priyadarshi, R., Gupta, B., Anurag, A.: Deployment techniques in wireless sensor networks: a survey, classification, challenges, and future research issues. J. Supercomput. **76**(9), 7333–7373 (2020)
13. Amutha, J., Sharma, S., Nagar, J.: WSN strategies based on sensors, deployment, sensing models, coverage and energy efficiency: review, approaches and open issues. Wireless Pers. Commun. **11**(2), 1089–1115 (2020)
14. Batalin, M.A., Sukhatme, G.S.: The analysis of an efficient algorithm for robot coverage and exploration based on sensor network deployment. In: Proceedings of the 2005 IEEE International Conference on Robotics and Automation, pp. 3478–3485 (2005)
15. Haotian, L., Barnawi, A., Stojmenovic, I., Wang, C.: Market-based sensor relocation by robot team in wireless sensor networks. Ad-Hoc Sens. Wirel. Netw. **22**(3), 259–280 (2014)

16. Wang, Y., Barnawi, A., Fernandes De Melle, R., Stojmenovic, I.: Localized ant colony of robots for redeployment in wireless sensor networks. Multi-Valued Logic Soft Comput. **23**, 35–51 (2014)
17. Desjardins, B., Falcon, R., Abielmona, R., Petriu, E.: A multi-objective optimization approach to reliable robot-assisted sensor relocation. In: IEEE Congress on Evolutionary Computation (CEC), pp. 56–64 (2015)
18. Desjardins, B., Falcon, R., Abielmona, R., Petriu, E.: Reliable multiple robot-assisted sensor relocation using multi-objective optimization. In: IEEE Congress on Evolutionary Computation (CEC), pp. 4476–4485 (2016)
19. Desjardins, B., Falcon, R., Abielmona, R., Petriu, E.: Planning robust sensor relocation trajectories for a mobile robot with evolutionary multi-objective optimization. In: Computational Intelligence in Wireless Sensor Networks, pp. 179–210 (2017)
20. Falcon, R., Nayak, A., Abielmona, R.: An evolving risk management framework for wireless sensor networks. In: IEEE International Conference on Computational Intelligence for Measurement Systems and Applications (CIMSA), pp. 1–6 (2011)
21. Cheng, C., Chen, Y., Lin, J.C.: A carrier-based sensor deployment algorithm for perception layer in the IoT architecture. IEEE Sens. J. **20**(17), 10295–10305 (2020)
22. Belguerche, N., Lasla, N., Benchaiba, M.: On optimal robot displacement for efficient coverage in WSN. In: The 15th International Conference on Advances in Mobile Computing and Multimedia, pp. 175–181 (2017)
23. Kun, M., Hailan, D., Feng, Q., Ye, D.: A One-Commodity pickup-and-delivery traveling salesman problem solved by a two-stage method : a sensor relocation application. PloS One **14**(4), e0215107 (2019)
24. Parretta, A.: All the light from the Newton's prism: effects of the multiple internal reflections. Opt. Commun. **474**, 126101 (2020)
25. Xiangyu, M., Yusheng, Z., Jitao, J., Zhipeng, W., Qilong, W.: The optical properties of a visible light filter integrated on the silicon substrate. Opt. Commun. **464**, 125510 (2020)
26. Taylor, R.L., Patterson, R.F., Bozorgnia, A.: Weak laws of large numbers for arrays of Rowes negatively dependent random variables. Appl. Math. Stoch. Anal. **86**(5), 804076 (2001)
27. Carbone, R., Girotti, F., Melchor-Hernandez, A.: On a generalized central limit theorem and large deviations for homogeneous open quantum walks. J. Stat. Phys. **188**(1), 8 (2022)
28. Horng-Jinh, C., Kuo-Chung, H., Chao-Hsien, W.: Determination of sample size in using central limit theorem for Weibull distribution. Int. J. Inf. Manage. Sci. **17**, 31 (2006)
29. Sakil, C., Benslimane, S.: Relocating redundant sensors in randomly deployed wireless sensor networks. In: IEEE Global Communications Conference (GLOBECOM), pp. 1–6 (2018)
30. Kusuma, M., Veena, N., Aparna, N.: Effective deployment of sensors in a wireless sensor networks using Hebbian machine learning technique. In: International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), pp. 268–274 (2021)
31. Biradar, S., Shastry, M.: Redundancy elimination with coverage preserving algorithm in wireless sensor network. Int. J. Commun. Netw. Inf. Secur. (IJCNIS) **10**(3), 454 (2018)
32. El Khamlichi, Y., Mesmoudi, Y., Tahiri, A., Abtoy, A.: A recovery algorithm to detect and repair coverage holes in wireless sensor network system. J. Commun. **13**(2), 67–74 (2018)
33. Feng, X., Zhang, X., Zhang, J., Muhdhar, A.A.: A coverage hole detection and repair algorithm in wireless sensor networks. Cluster Comput. **22**(22), 12473–12480 (2019)
34. Koriem, S.M., Bayoumi, M.A.: Detecting and measuring holes in wireless sensor network. J. King Saud Univ. Comput. Inf. Sci. **32**(8), 909–916 (2020)

Nadia Belguerche received the Computer Science Engineer Degree from the University of Sciences and Technologies Houari Boumedi `ene (USTHB) at Algiers in 2005. She received the Master Degree in computer science at USTHB in 2011. She is actually PHD student in the Computer Science Department at USTHB. Her research interests include Distributed Systems, Mobile Ad hoc Networks, Wireless sensor and robot net-works, Internet of Things, Industrial Wireless Sensor Network, Cyber physical system, Dis-tributed Algorithms and services.

Samir Brahim Belhaouari (Senior Member, IEEE) received the master's degree in Telecommunications from the National Polytechnic Institute (ENSEEIHT) of Toulouse, France, in 2000, and the Ph.D. degree in applied mathematics from the Federal Polytechnic School of Lausanne (EPFL), in 2006. He is presently an Associate Professor at the Division of Information and Communication Technologies, College of Science and Engineering, HBKU. Several academic and administrator positions are also held by him. These include the vice dean for academic and student affairs with the College of Science and General Studies and University Preparatory Program at ALFAISAL University (KSA), University of Sharjah (UAE), Innopolis University, Russia, Petronas University(Malaysia), and EPFL Federal Swiss School, Switzerland. Machine learning and number theory are his main research interests, with a proclivity to stochastic processes. He is currently involved in the development of algorithms in machine learning, applied to visual surveillance and biomedical data. He has also acquired the support of several international research funds in Russia, Malaysia, and GCC.

Noureddine Lasla received the B.Sc. and M.Sc. degrees in 2005 and 2008 from the University of Science and Technology Houari Boumediene (USTHB) and the Superior Computing National School (ESI), respectively, and received his Ph.D. degree in 2015 from the USTHB, all in computer science. He was a Researcher at CERIST research center, Algiers, Algeria from 2010 to 2017, and then a Research Scientist with the Qatar Mobility Innovations Center (QMIC), Doha, Qatar, from 2017 to 2018. Since 2018, he has been a Postdoctoral Research Fellow in the Division of Information and Computing Technology at Hamad Bin Khalifa University, Qatar with expertise in distributed systems, network communication, and cyber security.

Mahfoud Benchaïba received the Computer Science Engineer Degree from the University of Batna, Algeria in 1988. He received the Master Degree in computer science at the University of Sciences and Technologies Houari Boumediene (USTHB) at Algiers in 1992. He received the PhD Degree in Computer Science from the USTHB in November 2, 2009. He is actually full associate professor in the Computer Science Department at USTHB. He is a Research Director and He is in charge a research project on P2P systems in LSI Laboratory at USTHB. His research interests include Distributed Systems, Mobile Ad hoc Networks, Wireless sensor networks, Wireless body area networks, Distributed Algorithms and services, P2P systems.

Authors and Affiliations

Nadia Belguerche[1] · Samir Brahim Belhaouari[2] · Noureddine Lasla[3] · Mahfoud Benchaïba[1]

✉ Nadia Belguerche
nbelguerche@usthb.dz

Samir Brahim Belhaouari
sbelhaouari@hbku.edu.qa

Noureddine Lasla
Noureddine.lasla@ensia.edu.dz

Mahfoud Benchaïba
mbenchaiba@usthb.dz

[1] Department of Computer Science, University of Sciences and Technology, Houari Boumediene (USTHB), Algiers, Algeria

[2] Division of Information and Computing Technology College of Science and Engineering, Hamad Bin Khalifa University, Doha, Qatar

[3] The National School of Artificial Intelligence (ENSIA), Algiers, Algeria