Journal of Computer and Communications, 2026, 14(2), 39-63
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

AC-COA: A Coati-Inspired Optimization Algorithm for Area Coverage Enhancement in Wireless Sensor Networks

Pangop Doris-Khöler Nyabeye*, Ngangmo Olga Kengni

Department of Information and Communication Technologies, Faculty of Sciences, University of Ebolowa, Ebolowa, Cameroon
Email: *nyabeyedoriskholer@gmail.com

How to cite this paper: Nyabeye, P.D.-K. and Kengni, N.O. (2026) AC-COA: A Coati-Inspired Optimization Algorithm for Area Coverage Enhancement in Wireless Sensor Networks. *Journal of Computer and Communications*, **14**, 39-63.
https://doi.org/10.4236/jcc.2026.142003

Received: January 19, 2026
Accepted: February 11, 2026
Published: February 14, 2026

Abstract

Optimizing area of interest (AoI) coverage in wireless sensor networks (WSNs) is essential for ensuring reliable monitoring and high quality of service. Although numerous approaches have been proposed to enhance sensor coverage, many of them struggle to simultaneously achieve high coverage rates, efficient sensor distribution, and low computational cost. As a result, maximizing coverage in WSNs remains a challenging optimization problem that continues to attract significant research interest. In this paper, a novel area coverage approach based on the Coati Optimization Algorithm (COA), a recently proposed parameter-free meta-heuristic, is introduced. The proposed method, called AC-COA, enhances the original COA by incorporating adaptive sensor placement and diversification mechanisms that help avoid local optima and rapidly guide sensors toward under-covered regions. During the exploration phase, a scaling coefficient is employed to intensify the search in low-coverage areas. In the exploitation phase, a dynamic attenuation coefficient gradually reduces randomness, while a gradient-based overlap minimization term improves sensor distribution by limiting redundant coverage. The performance of AC-COA is evaluated through extensive simulations and compared with Particle Swarm Optimization (PSO), Improved Grey Wolf Optimization (IGWO), Improved Honey Badger Algorithm (IHBA), and Genetic Algorithm (GA) in terms of coverage rate and execution time. Simulation results demonstrate that AC-COA achieves superior coverage while significantly reducing sensor redundancy and computational cost. For a network consisting of 30 sensors, AC-COA outperforms GA by 15%, PSO and IGWO by 25%, and IHBA by 36%. Overall, AC-COA provides an effective solution for WSN area coverage optimization, offering an excellent trade-off between coverage efficiency and execution speed. Its parameter-free design makes it a robust and practical approach for applications requiring dense and continuous monitoring.

Keywords

Area Coverage Problem, Coati Optimization Algorithm, Swarm Intelligence, Wireless Sensor Networks

1. Introduction

A wireless sensor network (WSN) consists of a large number of interconnected sensor nodes that collaboratively sense, collect, and transmit data related to physical or environmental phenomena. Due to their flexibility and low deployment cost, WSNs have been widely adopted in many application domains, including environmental monitoring, industrial control, security surveillance, healthcare systems, and smart infrastructures. Despite their advantages, WSNs face several critical challenges, among which the coverage problem is one of the most fundamental and widely studied.

The coverage problem in WSNs refers to the deployment and positioning of sensors in such a way that every point within a given area of interest (AoI) is effectively monitored [1]. Achieving optimal coverage is essential not only for ensuring timely and accurate event detection, but also for reducing sensing redundancy and unnecessary overlaps between sensors, which in turn helps extend network lifetime. Coverage optimization is therefore crucial for mission-critical applications such as environmental monitoring, disaster management, border surveillance, and military operations. In the literature, coverage problems in WSNs are generally classified into three main categories: area coverage, point coverage, and barrier coverage [2] [3]. Area coverage aims to monitor every point in the AoI, point coverage focuses on a set of specific targets, and barrier coverage is designed to detect intrusions across a protected region.

In this work, we focus on the area coverage problem in WSNs. Numerous approaches have been proposed to address this problem, and they can broadly be classified into deterministic methods and meta-heuristic-based approaches [4] [5]. Deterministic solutions often suffer from limited scalability and poor adaptability to dynamic network conditions. Consequently, meta-heuristic algorithms have emerged as powerful alternatives for solving the area coverage problem in WSNs. Since area coverage can be formulated as a complex optimization problem, meta-heuristics and hybrid optimization techniques are widely used to enhance convergence speed, improve solution quality, and avoid premature convergence to local optima [1]-[3].

A wide variety of meta-heuristic algorithms have been investigated in the literature for WSN area coverage optimization. These include Particle Swarm Optimization (PSO) [4] [5]-[8], Grey Wolf Optimization (GWO) [9]-[12], Artificial Bee Colony (ABC) [13], Genetic Algorithm (GA) [4] [14], Harris Hawks Algorithm (HHA) [15], Glowworm Swarm Optimization (GSO) [16], Honey Badger Algorithm (HBA) [17], Greedy Algorithms [18], Firefly Algorithm (FA) [19], among others.

The development of new meta-heuristic optimization algorithms, particularly bio-inspired ones, remains an active and rapidly evolving research area. Researchers continue to propose novel algorithms to address increasingly complex optimization problems. In this context, the Coati Optimization Algorithm (COA), recently introduced by Dehghani *et al.* [20], represents a promising addition to the family of swarm intelligence methods. COA is inspired by the social behavior and hunting strategies of coatis in their natural habitat.

Motivated by the potential of this recent optimization algorithm, this paper proposes a novel area coverage approach for WSNs based on COA. The proposed method, named AC-COA, aims to evaluate and enhance the capability of COA in solving the area coverage optimization problem in wireless sensor networks. To the best of our knowledge, this is the first work that adapts the Coati Optimization Algorithm specifically for area coverage improvement in WSNs. The main contributions of this study are summarized as follows:

- An original and efficient adaptation of the Coati Optimization Algorithm (COA) for solving the area coverage problem in wireless sensor networks;
- A dedicated exploration strategy designed to identify and improve coverage in under-covered regions of the AoI;
- An adaptive exploitation strategy that combines dynamic attenuation of randomness with a gradient-based overlap minimization mechanism to optimize sensor placement;
- A comprehensive performance evaluation of the proposed AC-COA approach using multiple metrics and comparisons with state-of-the-art meta-heuristic-based area coverage algorithms.

The remainder of this paper is organized as follows. Section 2 reviews related work on meta-heuristic-based solutions for the area coverage problem. Section 3 presents an overview of the Coati Optimization Algorithm. Section 4 details the proposed AC-COA approach. Simulation results and performance analysis are discussed in Section 5. Finally, conclusions and future research directions are provided at the end of the paper.

2. Related Works

Meta-heuristic optimization techniques have been extensively investigated to address the area coverage problem in wireless sensor networks (WSNs). Among these techniques, Particle Swarm Optimization (PSO) is one of the most widely adopted algorithms, either used independently or hybridized with other meta-heuristics. In [5], PSO is combined with Differential Evolution to improve area coverage while extending network lifetime. Although this hybrid approach achieves improved monitoring performance, it significantly increases computational complexity and exhibits high sensitivity to the parameter settings of both algorithms. Kou *et al.* [6] propose a hybrid method named GWPSO, which integrates PSO with the Grey Wolf Optimization (GWO) algorithm. This approach achieves better coverage efficiency and network stability compared to standard PSO and GWO. However,

the hybridization leads to longer execution times than simpler optimization methods. In [7], PSO is combined with chaos optimization to enhance exploration capabilities through chaotic behavior. While this strategy improves coverage and network efficiency, it introduces additional computational overhead, resulting in increased processing time and complexity. The work presented in [8] evaluates PSO-based coverage optimization through extensive simulations under various network scenarios. Although the results demonstrate improved coverage performance, the approach remains highly dependent on parameter tuning, which limits its robustness and practical applicability.

Grey Wolf Optimization (GWO) has also been widely explored for coverage optimization in WSNs. Shipeng *et al.* [9] propose a coverage enhancement approach that integrates a Virtual Force Algorithm with a Lévy-flight-based GWO to improve sensor deployment efficiency. In [10], an Adaptive Learning Grey Wolf Optimizer (AL-GWO) is introduced to dynamically adjust algorithm parameters and enhance coverage performance. Despite its effectiveness, AL-GWO is difficult to implement and tune, and it may suffer from overfitting in specific scenarios, reducing its generalization capability. Yong *et al.* [11] combine GWO with Simulated Annealing to improve coverage efficiency and overall network performance. Similarly, Zhendong *et al.* [12] propose modifications to the standard GWO to improve convergence speed, search accuracy, and the ability to avoid local optima. However, these enhancements significantly increase algorithmic complexity, making implementation and parameter tuning more challenging.

Other swarm-based and evolutionary algorithms have also been investigated. In [13], the Bee Algorithm is applied to improve area coverage by optimizing sensor node deployment. While the algorithm demonstrates good adaptability and coverage performance across different network configurations, it introduces considerable complexity in terms of implementation and parameter tuning. Genetic Algorithms (GA) are used in [14] to maximize area coverage through evolutionary optimization of sensor placement. Although GA achieves satisfactory coverage improvement, it is computationally intensive, particularly for large-scale networks, due to its iterative evolutionary processes.

Harris Hawks Optimization (HHO) has been employed to enhance coverage while minimizing energy consumption in WSNs. Hakeem *et al.* [15] use HHO to optimize sensor deployment, achieving improved coverage efficiency. However, the method requires substantial computational resources, especially for large networks, which negatively impacts execution time. Glowworm Swarm Optimization (GSO) is explored in [16], where Voronoi diagrams, GSO, and K-Means clustering are combined to improve coverage quality through optimized node distribution and clustering. The integration of multiple techniques increases algorithmic complexity and makes implementation and parameter tuning more difficult.

More recently, Nguyen *et al.* [17] propose an improved Honey Badger Algorithm (IHBA) to optimize sensor node placement by exploiting the behavioral characteristics of honey badgers. Although the approach introduces an innovative

solution to coverage optimization, it still suffers from slow convergence and susceptibility to local optima.

Table 1 summarizes the main meta-heuristic-based approaches proposed for area coverage optimization in WSNs, highlighting their core techniques and principal limitations.

Table 1. Overview of related works on area coverage in WSNs using meta-heuristics.

Paper Reference	Meta-heuristic	Limitations
Farooq *et al.* (2018) [5]	PSO	Low coverage, local optima, high parameter sensitivity.
Wang *et al.* (2019) [9]	GWO	Slow convergence speed, low search precision, long execution time.
Wang *et al.* (2019) [12]	Improved GWO	Increased complexity, risk of local optima.
Hanh *et al.* (2019) [14]	GA	High computational complexity, unstable performance.
Zhang *et al.* (2020) [8]	Improved PSO	High computational cost, long execution time.
Zhang *et al.* (2021) [11]	GWO-SA	Non-uniform sensor distribution, slow convergence.
Zhao *et al.* (2022) [7]	PSO with chaos optimization	Premature convergence, high complexity in high-dimensional spaces.
Kou *et al.* (2023) [6]	Hybrid PSO-GWO	High computational complexity, long execution time.
Yang *et al.* (2023) [15]	HHA	Improper sensor deployment, high computational demand.
Nguyen *et al.* (2023) [17]	IHBA	Slow convergence, local optimum trapping.
Yu *et al.* (2024) [10]	Adaptive Learning GWO	High complexity, long execution time, overfitting risk.

PSO (Particle Swarm Optimization)—GWO (Grey Wolf Optimization)—IGWO (Improved Grey Wolf Optimization)—GA (Genetic Algorithm)—GWO-SA (Grey Wolf Optimization combined with Simulated Annealing)—HHA (Harris Hawks Algorithm)—IHBA (Improved Honey Badger Algorithm).

Overall, meta-heuristic-based approaches have demonstrated strong potential for solving the area coverage problem in WSNs. However, many existing methods suffer from high computational complexity, parameter sensitivity, and limited scalability. In this work, we address the area coverage problem using a recently proposed optimization technique known as the Coati Optimization Algorithm (COA).

Unlike many traditional meta-heuristics, COA is a parameter-free algorithm, which significantly reduces sensitivity to parameter tuning. Parameter-dependent meta-heuristics often require careful calibration to achieve optimal performance,

and their effectiveness may vary considerably across different problem instances. This dependence reduces their adaptability and increases implementation complexity. In contrast, COA avoids these limitations, making it a robust and promising candidate for coverage optimization in WSNs. These characteristics strongly motivate the investigation of COA for solving the area coverage problem.

3. Overview of the Coati Optimization Algorithm (COA)

The Coati Optimization Algorithm (COA) is a bio-inspired, population-based meta-heuristic proposed by Dehghani *et al.* [20]. This algorithm is inspired by the foraging and social behaviors of coatis, a species of mammals known for their cooperative hunting strategies. In COA, each individual coati represents a candidate solution to the optimization problem, while the entire population collaboratively explores the search space to identify the optimal solution according to predefined objective criteria.

Coatis are omnivorous animals, and one of their preferred prey is the green iguana, a large lizard commonly found in trees. During hunting, coatis adopt a cooperative strategy: some individuals climb trees to frighten the iguana and force it to jump to the ground, where other coatis are positioned to capture it. At the same time, coatis must remain vigilant against potential predators and may need to rapidly escape when threatened. These natural behaviors illustrate a dynamic balance between cooperative exploration and rapid, targeted responses.

Dehghani *et al.* demonstrated that the hunting strategy of coatis during iguana pursuit corresponds to an effective exploration mechanism, while their behavior when detecting and escaping predators represents an exploitation process. These two intelligent behaviors form the conceptual foundation of the COA design. The algorithm is primarily intended to solve complex optimization problems by maintaining a balance between exploration of the search space and exploitation of promising solutions, which are both essential for efficient optimization. By leveraging cooperative search patterns and adaptive movement strategies, COA effectively avoids premature convergence to local optima and accelerates the discovery of high-quality global solutions. The overall procedure of the COA algorithm can be summarized by the following pseudo-code:

1) **Initialize:** Set algorithm parameters like population size and maximum number of iterations, etc. Create the initial population of N coatis. Each coati is a candidate solutions having M dimension, and is generated by Equation (1)

$$x_{ij} = x_j^{\min} + r \cdot \left(x_j^{\max} - x_j^{\min} \right), i = 1, 2, \cdots, N \text{ and } j = 1, 2, \cdots, M \tag{1}$$

where x_{ij} is the position of the i-th coati in the j-th dimension. r is a random real number in [0, 1]. $x_j^{\min}$ and $x_j^{\max}$ are respectively the lower and upper bounds of the j-th dimension of the solution space.

2) **Evaluate Fitness:** Evaluate the fitness of each candidate solution.

3) **Loop Until Termination Criteria (maximum number of iterations or satisfactory fitness value) is Met:**

- **Exploration phase**. Iguana are assumed to occupy the ideal place within the population. First half of coatis climb on the tree to attack iguana. Then, the iguana jumps to a random position. The Second half of coatis wait on ground to hunt iguana when it falls from the tree. Equation (2) is used to update the position of the coatis that climb on the tree and Equation (3) is used to update the position of the coatis that wait on ground:

$$\mathbf{x}_{ij}^{(t+1)} = \mathbf{x}_{ij}^{(t)} + r \cdot \left(\mathbf{iguana}_j - I \cdot \mathbf{x}_{ij}^{(t)} \right), i = 1, 2, \cdots, N/2 \text{ and } j = 1, 2, \cdots, M \quad (2)$$

$$\mathbf{x}_{ij}^{(t+1)} = \begin{cases} \mathbf{x}_{ij}^{(t)} + r \cdot \left(\mathbf{iguana}_j^{(G)} - I \cdot \mathbf{x}_{ij}^{(t)} \right), & \text{if } F\left(\mathbf{iguana}^{(G)}\right) < F(i) \\ \mathbf{x}_{ij}^{(t)} + r \cdot \left(\mathbf{x}_{ij}^{(t)} - \mathbf{iguana}_j^{(G)} \right), & \text{otherwise} \end{cases} \quad (3)$$

where:

- $\mathbf{iguana}$ is the position of the best member of the population.
- $\mathbf{iguana}^{(G)}$ is the random position of the iguana when it falls to the ground: $\mathbf{iguana}_j^{(G)} = x_j^{\min} + r \cdot \left(x_j^{\max} - x_j^{\min} \right)$.
- I is an integer randomly selected from the set $[1,2]$.
- r is a random real number in $[0,1]$.
- $i = N/2+1, N/2+2, \cdots, N$. for Equation (3).
- F is the objective function.

The new position of a coati is accepted only if its a better than its old position.

- **Exploitation phase**. It models the natural behavior of coatis when encountering predators and escaping from predators. Positions of coatis are updated using Equation (4).

$$\mathbf{x}_{ij}^{(t+1)} = \mathbf{x}_{ij}^{(t)} + (1-2r) \cdot \left(x_j^{\min_l} + r \cdot \left(x_j^{\max_l} - x_j^{\min_l} \right) \right) \quad (4)$$

where:

- $x_j^{\min_l} = x_j^{\min} / t$.
- $x_j^{\max_l} = x_j^{\max} / t$.
- $t = 1, 2, \cdots, T$ (maximum number of iterations).
- $i = 1, 2, \cdots, N$.
- $j = 1, 2, \cdots, M$.

The new position of a coati is accepted only if its a better than its old position.

- Save the best candidate solution found so far.

4) **Output the Best Solution**

An analysis of the Coati Optimization Algorithm (COA) proposed by Dehghani *et al.* reveals several distinctive characteristics that can be considered significant advantages over many existing optimization methods:

- First, COA is formulated without control parameters. As a result, there is no need for extensive parameter tuning to achieve satisfactory performance, which simplifies its practical application.
- Second, COA has demonstrated strong effectiveness in solving a wide range of optimization problems across different scientific domains, including complex and high-dimensional problems. This capability is largely attributed to its pa-

rameter-free design.

- Third, COA exhibits an excellent balance between exploration and exploitation mechanisms, enabling fast convergence toward high-quality solutions.
- Finally, COA has shown strong performance in addressing real-world optimization problems, where robustness and adaptability are essential.

These characteristics clearly motivate our interest in adopting COA for solving optimization problems in wireless sensor networks, particularly for maximizing area coverage. In addition, the algorithm is straightforward to implement and easy to understand, which further supports its suitability for practical deployment.

4. Materials and Methods

4.1. Assumption

In the development of our work, we made some assumptions:

- Homogeneity of sensors: all sensors have the same properties.
- Grid-based environment: Area of interest (AoI) is modeled as a grid of cells. The grid model simplifies coverage by dividing the area into discrete cells, allowing algorithms to check if a sensor covers a specific area. This is useful for simulations and testing, as it reduces the complexity of continuous space.

A sensor in our WSN has a sensing range denoted R_s and a communication range denoted R_c. The sensing range defines the local area that the sensor can monitor, and the communication range defines its signal exchange range within the network.

4.2. Coverage Objective Function for Area Coverage (AC) Problem

Maximizing coverage consist to ensure that the largest possible portion of AoI is monitored by the sensors. It requires positioning the sensors in such way that their sensing ranges overlap minimally. Generally, maximal coverage is expressed by the percentage of total area covered by at least one active sensor.

In this work, we consider that our AoI has a width W_{AoI} and a height H_{AoI}. This AoI is fragmented into cells of the same size. Let W_{nc} be the number of cells in width and H_{nc} be the number of cells in height.

We use a probabilistic sensing model (PSM) to define the probability that a cell x in the AoI is covered by a sensor s. The probability is represented as a function $p(s,x)$, which is defined in three different cases depending on the distance between the sensor and the target cell, as given by Equation (5).

$$p(s,x)=\begin{cases}0, & r_s+r_e \le d(s,x)\\ e^{-\lambda\left(d(s,x)-(r_s-r_e)\right)^{\beta}}, & r_s-r_e < d(s,x) < r_s+r_e\\ 1, & r_s-r_e \ge d(s,x)\end{cases} \tag{5}$$

where:

- r_e: The sensing error, representing potential inaccuracies in sensor measurements.
- r_s: The sensing range of the sensor, indicating how far the sensor can detect.

- λ : The sensing attenuation coefficient, a parameter controlling how quickly the probability decreases with distance.
- β : A parameter that modulates the decay rate of the coverage probability with distance.

This coverage model, described in 3 cases how the probability of coverage decreases as the distance between the sensor and the target increases, taking into account both the sensor's range and any potential measurement errors. Figure 1 describes a look of a sensor in that coverage model.

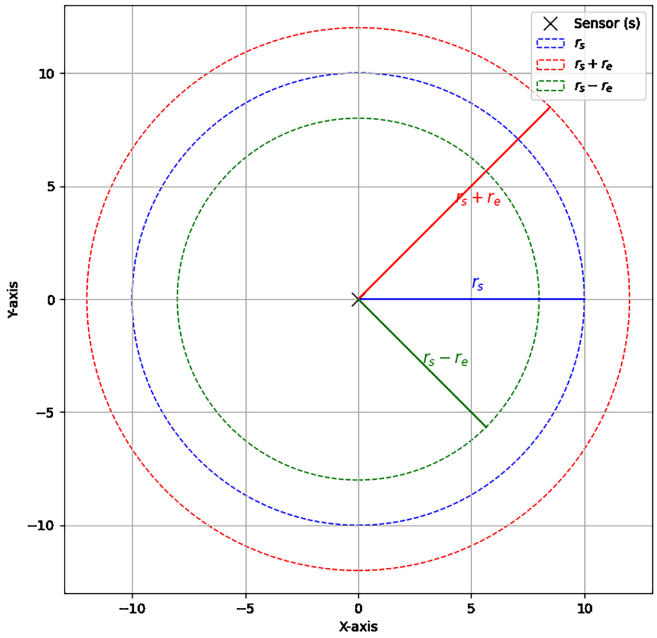


Figure 1. Sensor coverage model.

Thus, the probability p_x that x can be covered by the sensor network is defined by Equation (6).

$$p_x = 1 - \prod_{i=1}^{N} \left(1 - p\left(s_i, x\right)\right) \tag{6}$$

where N is the number of active sensor in the network.

To ultimately determine if a cell is covered or not in the area of interest (AoI), we use the following Equation (7).

$$Cov_{AoI}\left(x\right) = \begin{cases} 1, & p_x > p_\delta \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

where p_δ is a threshold necessary to determine if a cell is covered.

Let k be a possible sensors deployment configuration. The coverage objective function to maximize the coverage rate of the AoI by the sensor network is defined by Equation (8).

$$\text{Maximize } f\left(k\right) = \frac{\sum_{j=1}^{W_{nc} \times H_{nc}} \left(Cov_{AoI}\left(x_j^k\right)\right)}{W_{nc} \times H_{nc}} \tag{8}$$

W_{nc} is the number of cell along the width of the AoI and H_{nc} is the number of cell along the height of the AoI.

This formulation of the coverage objective involves minimizing uncovered areas.

4.3. Our Proposed Method: AC-COA

4.3.1. Coati and Its Population Representation

In AC-COA (Area Coverage based on Coati Optimization Algorithm), a coati is a possible configuration for node placement. Thus, a coati represents a solution of AC problem. Therefore, the size of a coati corresponds to the number of sensors in the network. Figure 2 represents an illustration of a coati in this work.

$S_1(X_1, Y_1)$	$S_2(X_2, Y_2)$	$\cdots$	$S_M(X_M, Y_M)$

$Msensors$

Figure 2. Coati representation in AC-COA.

Since COA is a population-based meta-heuristic, it requires an initial population of candidate solutions to operate. Let N be the size of the population and M be the number of sensors. A coati population is represented by the following Equation (9).

$$\mathbf{Pop} = \begin{bmatrix} C_1 \\ C_2 \\ \vdots \\ C_N \end{bmatrix} = \begin{bmatrix} S_{11}(X_{11}, Y_{11}) & S_{12}(X_{12}, Y_{12}) & \cdots & S_{1M}(X_{1M}, Y_{1M}) \\ S_{21}(X_{21}, Y_{21}) & S_{22}(X_{22}, Y_{22}) & \cdots & S_{2M}(X_{2M}, Y_{2M}) \\ \vdots & \vdots & \ddots & \vdots \\ S_{N1}(X_{N1}, Y_{N1}) & S_{N2}(X_{N2}, Y_{N2}) & \cdots & S_{NM}(X_{NM}, Y_{NM}) \end{bmatrix} \quad (9)$$

Each row of the matrix **Pop** corresponds to a potential configuration of sensor placements in the AoI.

4.3.2. Generation of the Initial Population

Like describes in Equation (9), each individual in the population is represented as a vector $\mathbf{C} = [S_1, S_2, \cdots, S_M]$, where S_j denotes the position of the j-th sensor in the deployment area. The AoI is modeled as a continuous 2D space $\mathcal{A} \subset \mathbb{R}^2$. The position of each sensor S_j is defined by Equation (10).

$$S_j = (x_j, y_j) \text{ where } x_j \in [x_{\min}, x_{\max}] \text{ and } y_j \in [y_{\min}, y_{\max}] \quad (10)$$

The initial population $\mathbf{Pop} = [\mathbf{C}_1, \mathbf{C}_2, \cdots, \mathbf{C}_N]$ is generated by LHS (Latin Hypercube Sampling) positions for each sensor within the defined search space. The i-th coati in **Pop** is represented by $\mathbf{C}_i = [S_{i1}, S_{i2}, \cdots, S_{iM}]$ (a sensor deployment configuration). Each sensor position S_{ij} for the i-th coati is generated by Equation (11):

$$S_{ij} = \begin{bmatrix} x_{ij} \\ y_{ij} \end{bmatrix} = \begin{bmatrix} x^{\min} + r_1 \cdot (x^{\max} - x^{\min}) \\ y^{\min} + r_2 \cdot (y^{\max} - y^{\min}) \end{bmatrix} \quad (11)$$

where r_1 and r_2 are uniformly distributed samples obtained via LHS over the interval $[0,1]$.

4.3.3. Evaluation of the Population

After generating the initial population, each coati is evaluated using the fitness function defined in Equation (8). This evaluation aims to quantify the proportion of discretized cells covered by the sensor placement configuration represented by each coati.

Once all coatis have been evaluated, the individual achieving the highest fitness value, corresponding to the maximum number of covered cells, is selected as the best coati. At each iteration of the algorithm, this best-performing coati represents the iguana, which serves as the reference solution guiding the search process.

4.3.4. Exploration Phase

In the proposed optimization framework, the group of coatis metaphorically represents the collective search process for solving the area coverage problem. Similar to the cooperative hunting behavior of coatis in nature, where individuals climb trees to scare iguanas into moving, the algorithm explores the search space by diversifying sensor placements. This exploration process enables the identification of poorly covered regions and plays a crucial role in preventing premature convergence to suboptimal solutions by preserving diversity among candidate configurations.

During this phase, half of the coatis are assigned to climb the tree, corresponding to an exploratory search toward higher or less-visited regions of the solution space, while the remaining coatis wait beneath the tree, ready to exploit newly discovered opportunities. When the iguana falls, its position is randomly relocated within the search space, reflecting the stochastic nature of the algorithm and enhancing its ability to adaptively search for optimal sensor configurations that maximize area coverage.

The position update of coatis climbing the tree is governed by Equation (12). Compared to the original formulation of this update rule see Equation (2), a scaling weight associated with low-coverage regions is introduced to encourage a more intensive exploration of under-covered areas. This modification improves exploration balance across the entire area of interest and enhances the algorithm's ability to identify high-quality coverage solutions.

$$\mathbf{x}_{ij}^{(t+1)} = \mathbf{x}_{ij}^{(t)} + r \cdot \left(\mathbf{iguana}_j - I \cdot \mathbf{x}_{ij}^{(t)} \right) \cdot w_j \tag{12}$$

where:

- $w_j = \dfrac{1}{1 + f\left(\mathbf{x}_i^{(t)}\right)}$ is a weight that encourages exploration of areas with low coverage.
- $\mathbf{iguana}$ is the position of the best member of the population.
- I is an integer randomly selected from the set $[1,2]$.
- r is a random real number in $[0,1]$.

- $i = 1, 2, \cdots, N/2$.
- $j = 1, 2, \cdots, M$.
- f is the objective function Equation (8).

Equation (13) is used to expressed the position of the coatis that wait on ground. Similarly, in the original version of the Equation of this phase Cf. Equation (3), we enhance the process of accepting new positions based on their contribution to the overall coverage by introducing the same scaling factor. Thus, we ensure that new positions are chosen not only if they are better (according to the fitness function) but also if they improve coverage in areas where there is still a gap.

$$\mathbf{x}_{ij}^{(t+1)} = \begin{cases} \mathbf{x}_{ij}^{(t)} + r \cdot \left(\mathbf{iguana}_j^{(G)} - I \cdot \mathbf{x}_{ij}^{(t)} \right) \cdot w_j, & \text{if } f\left(\mathbf{iguana}^{(G)}\right) < f\left(\mathbf{x}_i^{(t)}\right) \\ \mathbf{x}_{ij}^{(t)} + r \cdot \left(\mathbf{x}_{ij}^{(t)} - \mathbf{iguana}_j^{(G)} \right) \cdot w_j, & \text{otherwise} \end{cases} \tag{13}$$

where:

- $\mathbf{iguana}^{(G)}$ is the random position of the iguana when it falls to the ground: $\mathbf{iguana}_j^{(G)} = x_j^{\min} + r \cdot \left(x_j^{\max} - x_j^{\min} \right)$.
- w_j is a weight that encourages exploration of areas with low coverage.
- I is an integer randomly selected from the set $[1, 2]$.
- r is a random real number in $[0, 1]$.
- $i = N/2 + 1, N/2 + 2, \cdots, N$.
- $j = 1, 2, \cdots, M$.
- f is the objective function Equation (8).

When updating a coati, each new sensor position (each $x_j, j = 1, \cdots, M$) is accepted only if it is within the AOI. At the end of the update process, the new coati position obtained is accepted only if its fitness is better than the old coati fitness Cf. Equation (14).

$$\mathbf{x}_i^{(t+1)} = \begin{cases} \mathbf{x}_i^{(t+1)} & \text{if } f\left(\mathbf{x}_i^{(t+1)}\right) < f\left(\mathbf{x}_i^{(t)}\right) \\ \mathbf{x}_i^{(t)} & \text{otherwise} \end{cases} \tag{14}$$

4.3.5. Exploitation Phase

The exploitation phase of COA simulates the behavior of a coati when it encounters a predator or is threatened, prompting it to move toward a safe location. In the algorithm, this corresponds to generating new positions through a local search, keeping coatis close to their current locations while refining candidate solutions. In this work, we propose a more adaptive local search strategy that specifically avoids regions with overlapping sensor coverage, aiming to optimize sensor placement for maximal area coverage.

We enhance the original position update Equation for this phase, see Equation (4) by introducing the following modifications:

- First, a dynamic attenuation coefficient, $\alpha(t)$, defined in Equation (15), is used to progressively reduce random perturbations over iterations.
- Second, a gradient-based overlap minimization term, defined in Equation (17), is incorporated to guide sensors toward positions that reduce redundancy.

This gradient is scaled by a time-dependent weight, $\gamma(t)$, given in Equation (16), which increases over the course of the optimization.

This adaptive exploitation strategy enables broad exploration during early iterations to identify promising regions, followed by precise adjustments in later stages. By leveraging overlap gradients, the algorithm systematically reduces redundancy and produces efficient, non-overlapping sensor configurations. The approach thus achieves a dynamic transition from stochastic exploration to targeted optimization, significantly enhancing overall coverage efficiency.

$$\alpha^{(t)} = \left(1 - \frac{t}{T}\right) \tag{15}$$

$$\gamma^{(t)} = \gamma_{\text{init}} + \left(\gamma_{\text{final}} - \gamma_{\text{init}}\right) \cdot \frac{t}{T} \tag{16}$$

$$\nabla Overlap\left(\mathbf{x}_{ij}^{(t)}\right) = -\sum_{k \neq j} \frac{\mathbf{x}_{ij} - \mathbf{x}_{kj}}{\left\|\mathbf{x}_{ij} - \mathbf{x}_{ik}\right\|} \tag{17}$$

where:

- γ_{init} : Initial value of the gradient weight.
- γ_{final} : Final value of the gradient weight.
- $i = 1, 2, \cdots, N$ coatis.
- $j = 1, 2, \cdots, M$ sensors.

Equation (17) defines the gradient of overlap, denoted as $\nabla Overlap\left(x_{ij}\right)$, for each sensor in the deployment area. It is computed as the negative sum of unit vectors pointing from each neighboring sensor x_{ik} (where $k \neq j$) toward the sensor x_{ij}. This gradient indicates the direction in which the sensor j should move to increase its distance from its neighbors. The negative sign ensures that the gradient points in the direction that maximizes the separation between sensors, thereby reducing coverage overlap. By integrating this gradient into the position update process, the sensors are guided to disperse more evenly across the deployment area, which ultimately improves coverage efficiency by minimizing redundant sensing overlaps.

Finally, the updated position of a coati during exploitation phase is calculated using the following formula (18):

$$\mathbf{x}_{ij}^{(t+1)} = \mathbf{x}_{ij}^{(t)} + \alpha^{(t)} \cdot \left(1 - 2r\right) \cdot \left(x_j^{\min_l} + r \cdot \left(x_j^{\max_l} - x_j^{\min_l}\right)\right) - \gamma^{(t)} \cdot \nabla Overlap\left(\mathbf{x}_{ij}^{(t)}\right) \tag{18}$$

where:

- $x_j^{\min_l} = x_j^{\min} / t$.
- $x_j^{\max_l} = x_j^{\max} / t$.
- $t = 1, 2, \cdots, T$ (maximum number of iterations).
- $i = 1, 2, \cdots, N$ coatis.
- $j = 1, 2, \cdots, M$ sensors.
- r is a random real number in $[0,1]$.

When updating a coati, each new sensor position (each $x_j, j = 1, \cdots, M$) is accepted only if it is within the AOI. At the end of the update process, the new coati position obtained is accepted only if its fitness is better than the old coati fitness

Cf. Equation (14).

This targeted adjustment ensures that the final configurations achieve high coverage rates.

4.3.6. Algorithm of AC-COA

Algorithm of AC-COA method is given by Algorithm 1.

Algorithm 1: Algorithm of AC-COA

Input: Optimization problem information
Output: Best obtained solution by AC-COA : C^{best}

Input the optimization problem information;
Set T = number of iterations, N = number of coatis, and M = number of sensors, bounds of AoI ;
Population initialization by Eq. 10 and 11 ;
Proceed fitness evaluation of population ;
Choose coati with the best fitness value like the best member of population (C^{best}) ;
for $t = 1$ **to** T **do**
    Update location of the iguana based on C^{best};
    Phase 1: Exploration Phase;
    for $i = 1$ **to** $\lfloor N/2 \rfloor$ **do**
        Calculate new position for the ith coati using Eq. 12 ;
        Update its position using Eq. 14;
    end
    for $i = 1 + \lfloor N/2 \rfloor$ **to** N **do**
        Calculate random position for the iguana ;
        Calculate new position for the ith coati using Eq. 13;
        Update its position Eq. 14 ;
    end
    Phase 2: Exploitation Phase;
    for $i = 1$ **to** N **do**
        Calculate the new position for the ith coati using Eq. 18;
        Update its position Eq. 14;
    end
    Proceed fitness evaluation of population ;
    Choose coati with the best fitness value like the best member of population (C^{best}) ;
end
Output the best obtained solution C^{best} ;

5. Simulations and Interpretation of Results

5.1. Parameter Settings

The performance of AC-COA was evaluated through simulations and compared to other optimization algorithms used for coverage optimization in WSNs.

We compare our results with the following existing results in the literature:

- IHBA: Improved Honey Badger Algorithm [17].
- GA: Genetic Algorithm [14].
- PSO: Particle Swarm Optimization [8].
- IGWO: Improved Grey Wolf Optimization [12].

The comparison takes into account two important aspects of a meta-heuristic:

accuracy (given by average area coverage) and efficiency (given by running time).

Table 2 shows the global parameter settings for carrying out these simulations. For IHBA, GA, PSO, and IGWO, the parameter values are the same as those in the corresponding publications. We have implemented all algorithms in Java and run them on a machine with Intel Core i5 2.4 GHz, RAM 8 GB.

Table 2. Simulation parameters.

Parameters	Values
Environment size	500 × 500
Cell size	10 × 10
Number of sensors	10-20-30
Sensing range (R_s)	50
Coverage model	$\lambda = \frac{1}{R_s}, \beta = 2, r_e = 0.1 * R_s, P_\delta = 0.4$
Number of iterations	100
Population Size	100
γ_{init}	0.1
γ_{final}	0.5

Each result presented was obtained through 10 simulation runs to highlight the average performance of each algorithm.

5.2. Results and Discussions

Table 3 compares the proposed AC-COA method to other algorithms: IHBA [17], GA [14], PSO [8], IGWO [12]. Comparisons are done based on factors like coverage rate, running times, convergence iterations, and number of sensors.

Table 3. Comparison of AC-COA with the other algorithms in different cases such as coverage rate, running times, iterations to convergence and number of sensor nodes.

Algorithms	Factors	10 sensors	20 sensors	30 sensors
AC-COA	Coverage rate (%)	**35%**	**69%**	**90%**
	Running time(s)	**5,08E+00**	**7,92E+00**	**11,58E+00**
	Iterations to convergence	**30**	**57**	**55**
PSO	Coverage rate (%)	32%	54%	72%
	Running time(s)	3,61E+00	5,95E+00	8,79E+00
	Iterations to convergence	18	25	33
IGWO	Coverage rate (%)	34%	60%	72%
	Running time(s)	7,75E+00	12,40E+00	18,50E+00
	Iterations to convergence	70	82	80
IHBA	Coverage rate (%)	31%	50%	65%

Continued

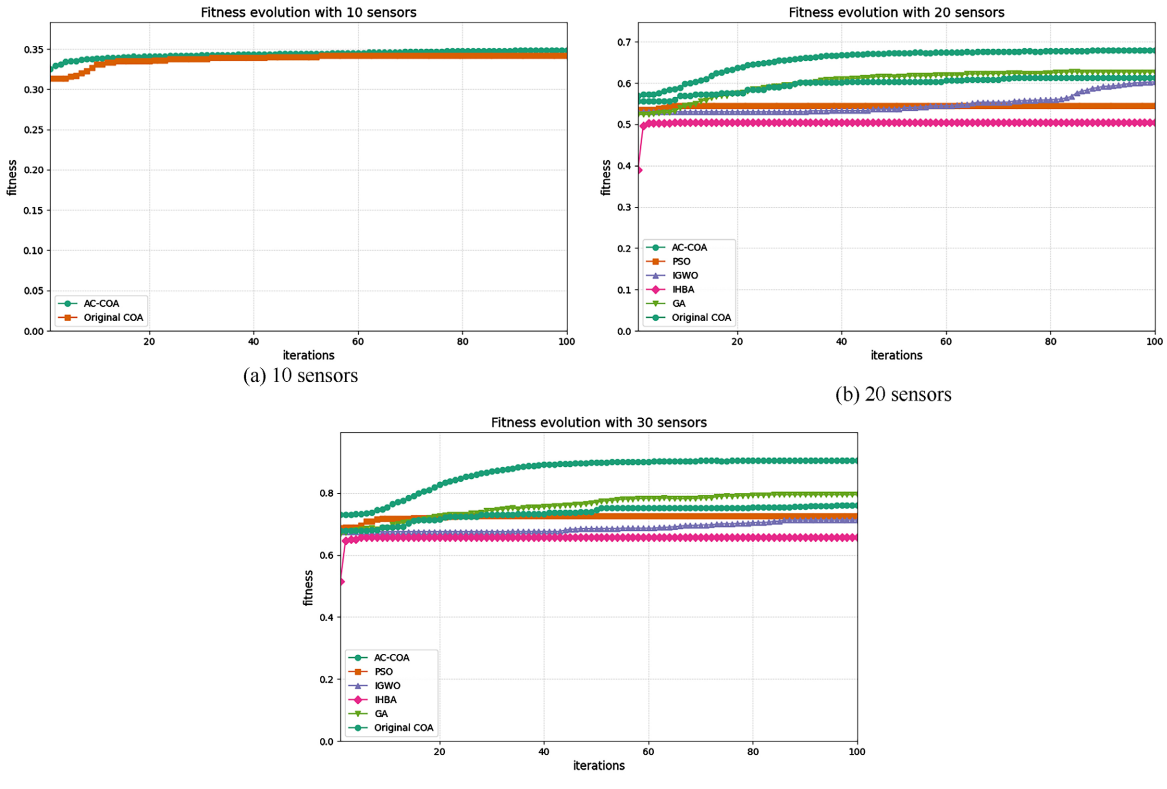
	Running time(s)	4,02E+00	6,12E+00	9,44E+00
	Iterations to convergence	22	30	30
	Coverage rate (%)	34%	62%	79%
GA	Running time(s)	8,68E+00	13,54E+00	21,66E+00
	Iterations to convergence	40	70	75

Globally, in Table 3, we notice that with 10, 20 or 30 sensors, AC-COA succeeds in covering the AoI better compared to the other approaches, since its coverage rate is each time higher than that of the other approaches evaluated.

5.2.1. Convergence

To better present the convergence performances of the studied methods compared to our proposed method, as mentioned in Table 3, we have highlighted in Figure 3 the evolution of the fitness function for each algorithm over the iterations.

The AC-COA algorithm also demonstrates faster convergence compared to the other algorithms. For 10 sensor nodes, AC-COA converges in 30 iterations, which is more efficient than GA (40 iterations) and IGWO (70 iterations), although slower than PSO (18 iterations) and IHBA (22 iterations). For 20 nodes, AC-COA



(a) 10 sensors

(b) 20 sensors

(c) 30 sensors

Figure 3. Average convergence of fitness function for each algorithm with different number of sensor nodes.

reaches convergence in 57 iterations, again outperforming GA (70 iterations) and IGWO (82 iterations), while being slower than PSO (25 iterations) and Original COA (40 iterations). For 30 nodes, AC-COA converges in 55 iterations, still faster than GA (75 iterations) and IGWO (80 iterations), but slightly slower than PSO (33 iterations) and IHBA (30 iterations).

We know that PSO is well-regarded for its fast convergence. However, in optimization algorithms, a rapid convergence often results in local minima, hindering the discovery of better solutions. Indeed, the rapid convergence can prevent the algorithm from exploring the search space thoroughly. Therefore, it can be noted that although a method converges faster than AC-COA, it does not obtain a better coverage rate, because it fails to efficiently explore the search space like AC-COA. On the other hand, we observed that IGWO faced difficulties in maintaining exploration and exploitation balance, which resulted in unstable convergence, especially in later iterations.

AC-COA does not converge very quickly because it maintains a good level between exploration and exploitation. But when we get to the end of the iterations, it stabilizes and produces a better result than the other methods studied.

These results show that AC-COA not only achieves better coverage but does so in fewer iterations, making it more efficient in terms of computational resources. This faster convergence is particularly advantageous in real-time or dynamic applications, such as surveillance or area coverage, where quick adaptation to changing environments or node positions is essential.

5.2.2. Coverage Rate

Figure 4 presents more clearly the results of Table 3 according to the coverage rate parameter. It can be observed that our AC-COA solution demonstrates significant improvements over the other evaluated methods. This further highlights the COA capability in solving the area coverage problem.

For 10 sensor nodes, AC-COA achieves 35% coverage, surpassing PSO (32%) and IGWO (34%) by 3% and 1%, respectively. For 20 nodes, AC-COA's coverage reaches 69%, a 7% increase over GA (62%) and 15% over PSO (54%). This corresponds respectively to 175 and 375 additional cells or areas covered in an AOI of 2500 cells, like in our simulations, highlighting the substantial gain in coverage. Finally, for 30 nodes, AC-COA covers 90% of the AOI, outperforming GA (79%) and PSO (72%) by 11% and 18%, respectively, further reducing the uncovered area and adding more than 275 additional covered cells. These results clearly demonstrate AC-COA's superior ability to optimize coverage, effectively decreasing the percentage of uncovered cells and increasing the total number of covered cells. In the context of coverage or surveillance applications, this improvement is crucial, as maximizing the coverage directly enhances the monitoring capability of the sensor network, ensuring more comprehensive and efficient surveillance of the area of interest with fewer sensor nodes, ultimately reducing the operational costs and improving reliability.

Table 4 presents the coverage performance of AC-COA compared to PSO,

IGWO, IHBA, and GA over 10 simulation runs for different numbers of sensors. For each algorithm, we report the coverage achieved in each run, the mean coverage with standard deviation (Avg ± Std), and the p-values obtained from the Wilcoxon rank-sum test comparing AC-COA with each competitor.

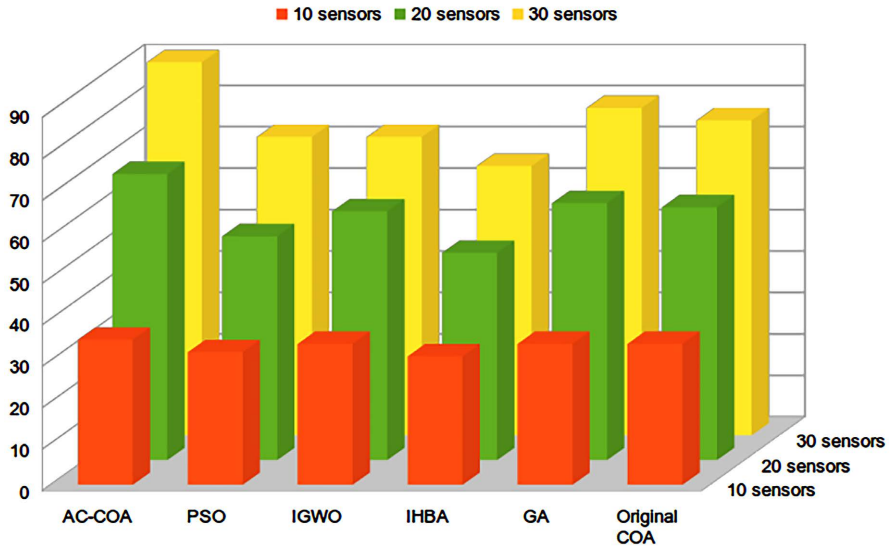


Figure 4. Coverage rate comparison.

The results indicate that AC-COA consistently achieves higher coverage across all scenarios. For instance, with 10 sensors, AC-COA attains an average coverage of 35.0% ± 0.84, significantly higher than PSO (32.0% ± 0.81, p = 0.000183) and IHBA (31.0% ± 0.41, p = 0.000182). Similarly, for 20 sensors, AC-COA reaches 69.0% ± 0.77, outperforming GA (62.0% ± 0.71, p = 0.000182) and PSO (54.0% ± 0.77, p = 0.000182). For 30 sensors, AC-COA achieves 90.0% ± 0.82, compared to GA (79.0% ± 0.77, p = 0.000181) and IGWO (72.0% ± 0.71, p = 0.000182).

Table 4. Coverage results for 10 simulation runs, mean ± standard deviation, and Wilcoxon rank-sum p-values comparing AC-COA to other algorithms.

Sensors	Algorithm	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	Avg ± Std	p-value
10	AC-COA	33.8	34.2	35.1	36.0	34.7	35.5	36.3	34.0	35.0	35.4	35.0 ± 0.84	-
	PSO	30.8	31.5	32.1	33.0	31.9	32.4	30.9	33.2	31.7	32.5	32.0 ± 0.81	0.000183
	IGWO	33.1	33.5	33.8	34.0	34.2	34.4	34.6	34.0	33.7	34.7	34.0 ± 0.50	0.009968
	IHBA	30.2	30.6	30.9	31.0	31.2	31.4	31.6	31.0	30.8	31.3	31.0 ± 0.41	0.000182
	GA	32.9	33.4	33.9	34.1	34.3	34.8	35.0	34.2	33.6	33.8	34.0 ± 0.63	0.015448
20	AC-COA	67.8	68.4	69.1	70.2	68.9	69.5	70.0	68.6	69.3	68.2	69.0 ± 0.77	-
	PSO	52.6	53.4	54.2	55.1	53.8	54.6	54.9	53.7	54.3	53.4	54.0 ± 0.77	0.000182
	IGWO	58.7	59.4	60.1	61.0	59.8	60.6	60.9	59.7	60.3	59.5	60.0 ± 0.72	0.000183
	IHBA	48.6	49.4	50.2	51.0	49.8	50.5	50.8	49.7	50.3	49.7	50.0 ± 0.71	0.000182
	GA	60.8	61.4	62.1	63.0	61.8	62.6	62.9	61.7	62.3	61.4	62.0 ± 0.71	0.000182

Continued

	AC-COA	88.6	89.3	90.1	91.2	89.8	90.7	91.0	89.7	90.3	89.3	90.0 ± 0.82	-
	PSO	70.6	71.4	72.2	73.1	71.8	72.6	72.9	71.7	72.3	71.4	72.0 ± 0.77	0.000181
30	IGWO	70.8	71.6	72.1	73.0	71.9	72.5	72.8	71.8	72.4	71.1	72.0 ± 0.71	0.000182
	IHBA	63.6	64.4	65.1	66.2	64.8	65.6	65.9	64.7	65.3	64.4	65.0 ± 0.78	0.000181
	GA	77.6	78.4	79.2	80.1	78.8	79.6	79.9	78.7	79.3	78.4	79.0 ± 0.77	0.000181

The low standard deviations demonstrate that AC-COA's performance is stable and robust across multiple runs. The Wilcoxon rank-sum test confirms that the improvements in coverage are statistically significant in all cases, with all p-values well below the 0.05 significance threshold, validating the reliability of AC-COA's superior performance. These results substantiate that AC-COA not only provides higher average coverage but also ensures consistent outcomes, making it a reliable and effective approach for area coverage optimization in wireless sensor networks.

The power of genetic algorithms has been widely used to optimize area coverage in WSNs. But during simulations, we noticed that the original COA had performances almost close to or very competitive with that of GA. However, AC-COA outperformed both COA and GA. This pushes us to confirm the idea that COA is a promising optimization method in the field of WSNs. This may also be partly due to the fact that COA is independent of any parameter. However, for parameter-related methods, the choice of parameters is crucial for obtaining good results, depending on the problem. But we can see that COA obtains very good results without the need to configure any parameter well.

5.2.3. Distribution of Nodes in the AOI

Figures 5-7 present an overview of the AoI coverage state provided at the end of the execution of each algorithm. We observe Figure 5 and Figure 6, that with 10 and 20 sensors, our solution provides coverage where the sensing ranges of the sensors do not overlap, thus demonstrating its effectiveness in striving to achieve the most extensive possible coverage of the AoI. When the number of sensors increases, as we can see in Figure 7, our algorithm provides coverage with less overlap of the sensing ranges than the other algorithms studied, and with a better distribution of sensors over the AoI. This confirms the relevance of COA and its effectiveness in achieving maximum area coverage by sensors with minimum overlap or non-overlap.

5.2.4. Running Time

Regarding execution time, Figure 8 shows that the AC-COA algorithm demonstrates competitive execution times compared to the other algorithms, balancing coverage and efficiency.

For 10 sensor nodes, AC-COA achieves a running time of 5.08 seconds, which is slightly slower than PSO (3.61 seconds) but faster than IGWO (7.75 seconds) and GA (8.68 seconds). For 20 nodes, AC-COA's execution time rises to 7.92 seconds, still outperforming IGWO (12.40 seconds) and GA (13.54 seconds) but

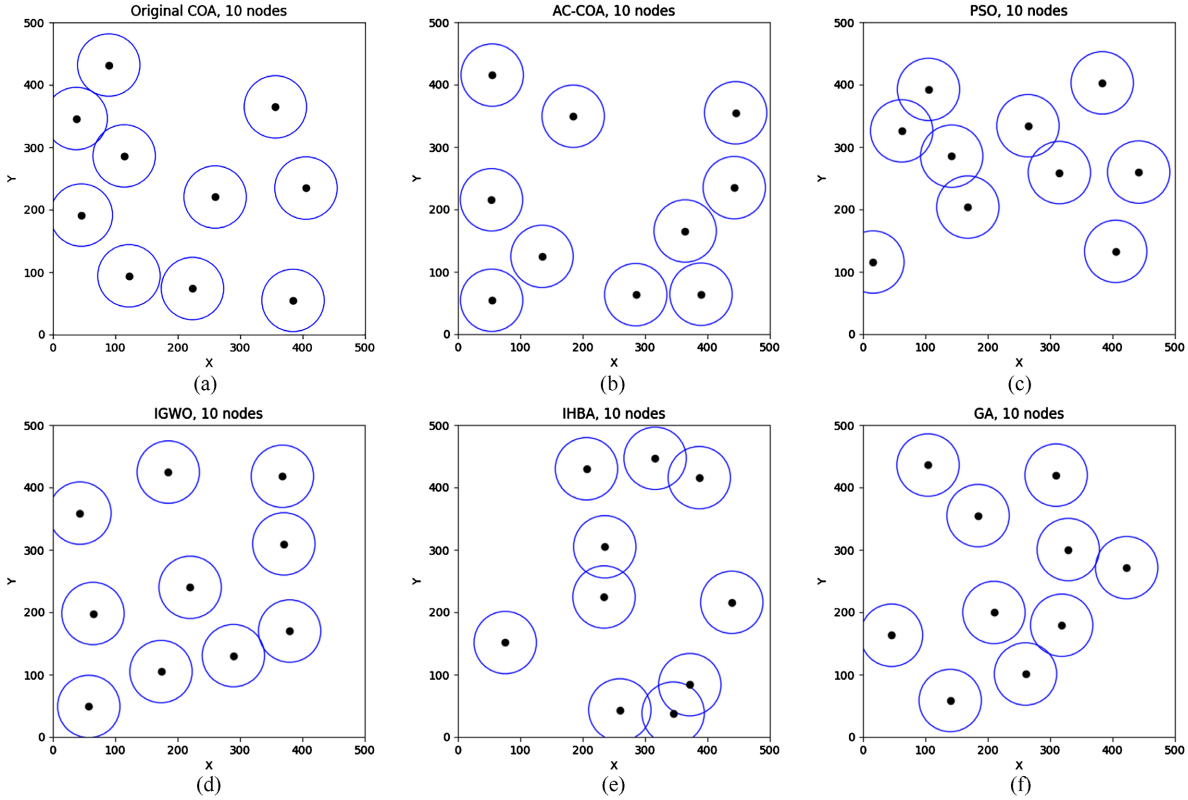


Figure 5. Area coverage with each algorithm using 10 sensors.

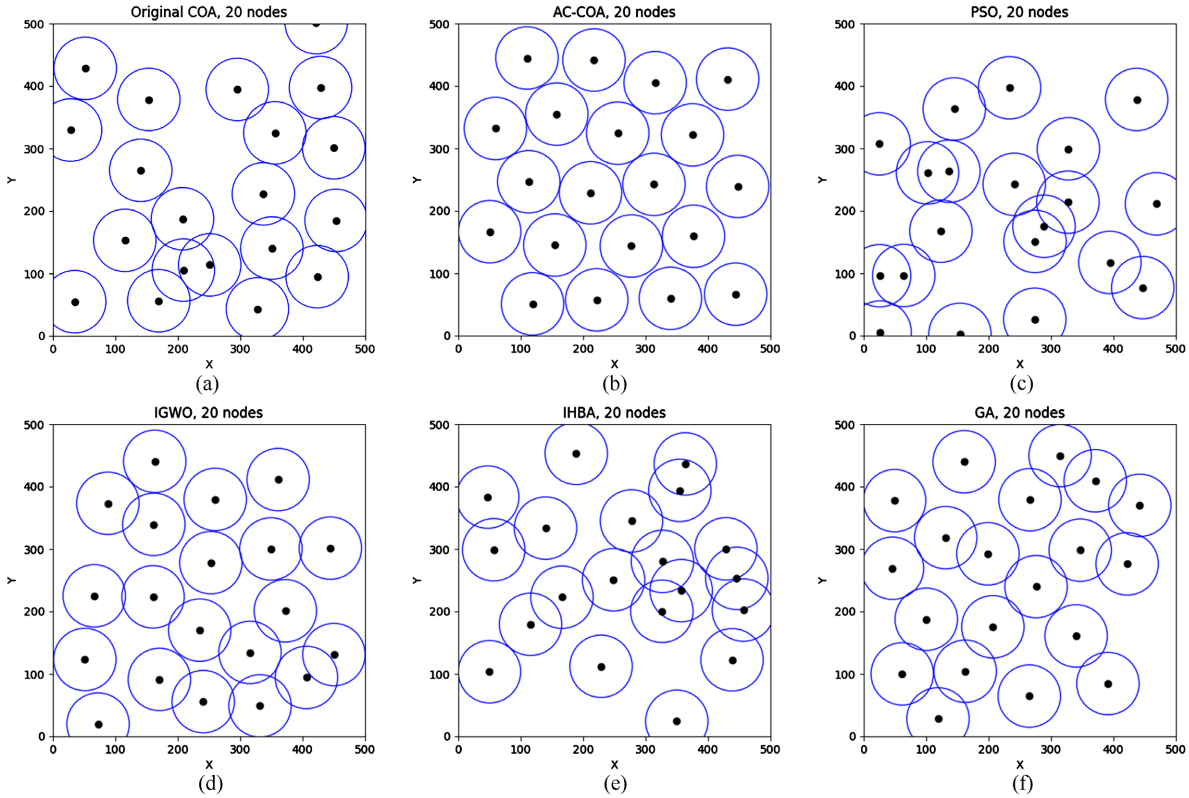


Figure 6. Area coverage with each algorithm using 20 sensors.

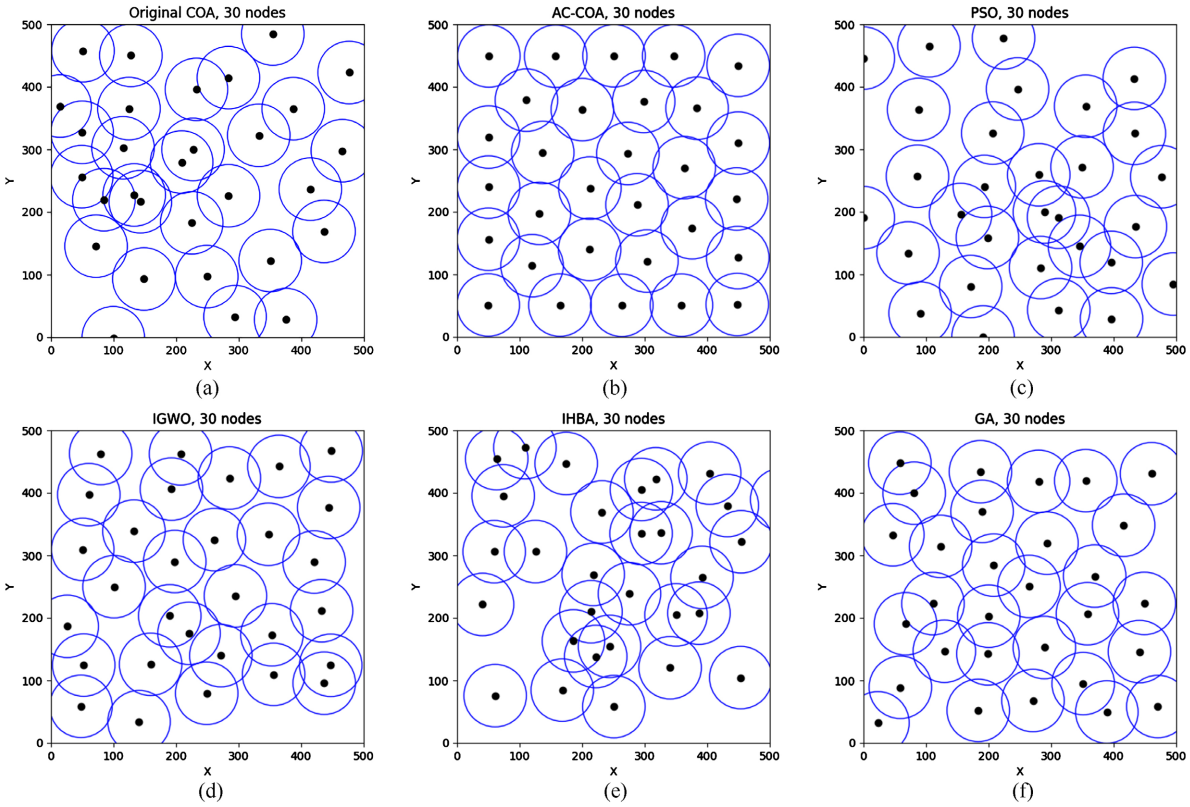


Figure 7. Area coverage with each algorithm using 30 sensors.

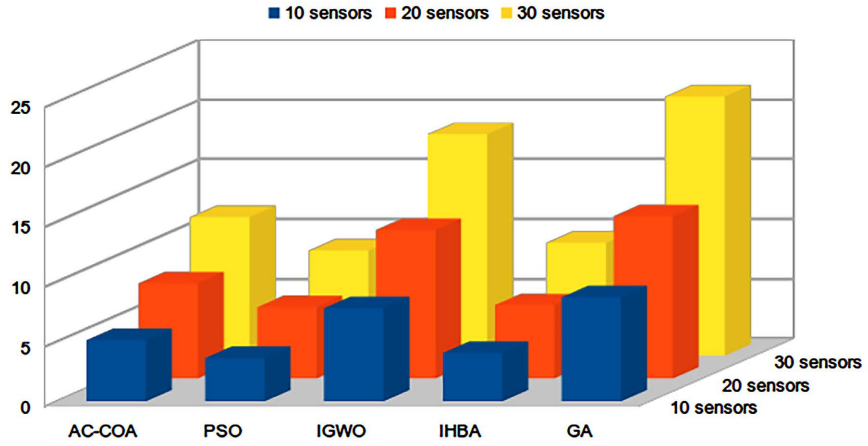


Figure 8. Running times comparison.

remaining slower than PSO (5.95 seconds). With 30 nodes, AC-COA completes in 11.58 seconds, which, while slightly slower than PSO (8.79 seconds), remains competitive compared to IGWO (18.50 seconds) and GA (21.66 seconds).

These results demonstrate that AC-COA achieves a favorable balance between high coverage performance and execution speed. The efficient design of AC-COA ensures that it is computationally feasible even for larger networks. Indeed, in terms of computational complexity, AC-COA's faster execution times relative to

IGWO and GA suggest that it handles larger solution spaces and fitness evaluations more efficiently, making it suitable for resource-constrained WSN applications. While slightly slower than PSO, AC-COA compensates by offering higher coverage rates, making it well-suited for time-sensitive scenarios, such as surveillance or environmental monitoring, where maintaining both computational efficiency and solution quality is critical.

5.2.5. Computational Complexity Analysis

The computational complexity of the proposed AC-COA algorithm can be analyzed based on its main operations. Let N denote the number of coatis (population size), M the number of sensors in the network, and T the number of iterations. The key steps contributing to the overall complexity are:

- Fitness Evaluation: For each coati, the coverage of the AoI is computed over $W_{nc} \times H_{nc}$ discretized cells. This leads to a complexity of $O\left(N \cdot M \cdot W_{nc} \cdot H_{nc}\right)$ per iteration.
- Exploration Phase Updates: Updating the positions of coatis during exploration involves simple arithmetic operations and random number generation, leading to $O\left(N \cdot M\right)$ per iteration.
- Exploitation Phase with Overlap Minimization: The adaptive local search incorporates the gradient of overlap between sensor positions. Computing the pairwise distances for M sensors requires $O\left(M^2\right)$ operations per coati. Thus, this step has a complexity of $O\left(N \cdot M^2\right)$ per iteration.

Therefore, the overall time complexity of AC-COA can be expressed as in Equation (19):

$$O\left(T \cdot \left(N \cdot M \cdot W_{nc} \cdot H_{nc} + N \cdot M + N \cdot M^2\right)\right) = O\left(T \cdot N \cdot \left(M \cdot W_{nc} \cdot H_{nc} + M^2\right)\right) \quad (19)$$

This analysis shows that the dominant term is the fitness evaluation over all cells, which grows linearly with the number of coatis and the number of sensors, but also with the total number of discretized cells. The pairwise overlap calculations add an additional $O\left(N \cdot M^2\right)$ term, which is negligible for small to moderate numbers of sensors compared to the cell-wise coverage evaluation.

Hence, despite the additional computations for overlap minimization, AC-COA maintains a practical computational cost, making it suitable for wireless sensor network deployment scenarios with real-time or near-real-time requirements.

5.3. Implications for Energy Consumption, Network Lifetime, and Scalability

Although the primary objective of this work is to maximize coverage, the proposed AC-COA approach also has important implications for energy consumption and network lifetime in wireless sensor networks. By explicitly reducing sensing overlap through the gradient-based repulsion mechanism, redundant sensing activities among closely located sensors are minimized, which helps avoid unnecessary energy expenditure. Moreover, the resulting more uniform spatial distri-

bution of sensors contributes to balanced energy consumption across the network, potentially mitigating premature energy depletion of specific nodes. While an explicit energy consumption model is beyond the scope of this study, the improved deployment achieved by AC-COA is expected to indirectly enhance network lifetime by reducing redundancy and promoting efficient resource utilization.

Additionally, it should be noted that the simulations presented in this work focus on small- to medium-sized WSNs (up to 30 sensors) to allow a detailed comparison with existing algorithms. While AC-COA has not yet been evaluated for very large-scale networks, its population-based structure and vectorized computations suggest that it can be extended to larger deployments. Future work will investigate the scalability of AC-COA for networks with hundreds or thousands of nodes, including optimizations such as parallelization, hierarchical deployment strategies, or distributed implementations to ensure computational efficiency while maintaining high coverage performance.

6. Conclusions

In this paper, we have proposed a novel method, AC-COA, to address the area coverage problem in wireless sensor networks (WSNs). The approach is based on the Coati Optimization Algorithm (COA), a bio-inspired meta-heuristic that models the cooperative foraging behavior of coatis to solve complex optimization problems. Although COA has been applied to other optimization domains, its adaptation to WSNs is particularly relevant due to the specific challenges of sensor deployment, including maximizing coverage while minimizing overlap.

To tailor COA for area coverage, we introduced key modifications to both the exploration and exploitation phases. The exploration strategy targets under-covered regions to ensure complete monitoring of the area of interest (AoI), while the exploitation phase reduces sensor redundancy, improving resource utilization. Simulation results demonstrate that AC-COA consistently achieves superior coverage compared to existing meta-heuristic approaches, including GA, PSO, IHBA, and IGWO, both for networks with few sensors and for larger deployments. Furthermore, the method exhibits fast convergence, reduced execution time, and stable performance across diverse test scenarios.

These findings highlight the effectiveness and robustness of COA as a foundation for WSN optimization. They also motivate future research directions, including the application of AC-COA to other WSN objectives such as node localization, connectivity, and energy efficiency, which constitute promising avenues for extending this work. As another future perspective, integrating an explicit energy consumption model into the AC-COA framework to jointly optimize coverage and network lifetime will be investigated, enabling a more comprehensive analysis of the trade-off between sensing performance and energy efficiency in WSN.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Elhabyan, R., Shi, W. and St-Hilaire, M. (2019) Coverage Protocols for Wireless Sensor Networks: Review and Future Directions. *Journal of Communications and Networks*, **21**, 45-60. https://doi.org/10.1109/jcn.2019.000005

[2] Amutha, J., Sharma, S. and Nagar, J. (2019) WSN Strategies Based on Sensors, Deployment, Sensing Models, Coverage and Energy Efficiency: Review, Approaches and Open Issues. *Wireless Personal Communications*, **111**, 1089-1115. https://doi.org/10.1007/s11277-019-06903-z

[3] Egwuche, O.S., Singh, A., Ezugwu, A.E., Greeff, J., Olusanya, M.O. and Abualigah, L. (2023) Machine Learning for Coverage Optimization in Wireless Sensor Networks: A Comprehensive Review. *Annals of Operations Research*. https://doi.org/10.1007/s10479-023-05657-z

[4] Binh, H.T.T., Hanh, N.T., Van Quan, L., Nghia, N.D. and Dey, N. (2020) Metaheuristics for Maximization of Obstacles Constrained Area Coverage in Heterogeneous Wireless Sensor Networks. *Applied Soft Computing*, **86**, Article ID: 105939. https://doi.org/10.1016/j.asoc.2019.105939

[5] Farooq, A. and Iqbal, T. (2018) An Exposition of Wireless Sensor Network Area Coverage and Lifetime Based on Meta Heuristic and Particle Swarm Optimization Algorithms. *VAWKUM Transactions on Computer Sciences*, **15**, 92. https://doi.org/10.21015/vtcs.v15i2.519

[6] Kou, G. and Wei, G. (2023) Hybrid Particle Swarm Optimization-Based Modeling of Wireless Sensor Network Coverage Optimization. *International Journal of Advanced Computer Science and Applications*, **14**, 982-991. https://doi.org/10.14569/ijacsa.2023.01405102

[7] Zhao, Q., Li, C., Zhu, D. and Xie, C. (2022) Coverage Optimization of Wireless Sensor Networks Using Combinations of PSO and Chaos Optimization. *Electronics*, **11**, Article No. 853. https://doi.org/10.3390/electronics11060853

[8] Zhang, Y. (2019) Coverage Optimization and Simulation of Wireless Sensor Networks Based on Particle Swarm Optimization. *International Journal of Wireless Information Networks*, **27**, 307-316. https://doi.org/10.1007/s10776-019-00446-7

[9] Wang, S., Yang, X., Wang, X. and Qian, Z. (2019) A Virtual Force Algorithm-Lévy-Embedded Grey Wolf Optimization Algorithm for Wireless Sensor Network Coverage Optimization. *Sensors*, **19**, Article No. 2735. https://doi.org/10.3390/s19122735

[10] Yu, X., Duan, Y., Cai, Z. and Luo, W. (2024) An Adaptive Learning Grey Wolf Optimizer for Coverage Optimization in WSNs. *Expert Systems with Applications*, **238**, Article ID: 121917. https://doi.org/10.1016/j.eswa.2023.121917

[11] Zhang, Y., Cao, L., Yue, Y., Cai, Y. and Hang, B. (2021) A Novel Coverage Optimization Strategy Based on Grey Wolf Algorithm Optimized by Simulated Annealing for Wireless Sensor Networks. *Computational Intelligence and Neuroscience*, **2021**, Article ID: 6688408. https://doi.org/10.1155/2021/6688408

[12] Wang, Z., Xie, H., Hu, Z., Li, D., Wang, J. and Liang, W. (2019) Node Coverage Optimization Algorithm for Wireless Sensor Networks Based on Improved Grey Wolf Optimizer. *Journal of Algorithms & Computational Technology*, **13**, 1-15. https://doi.org/10.1177/1748302619889498

[13] Khalaf, O.I., Abdulsahib, G.M. and Sabbar, B.M. (2020) Optimization of Wireless Sensor Network Coverage Using the Bee Algorithm. *Journal of Information Science and Engineering*, **36**, 377-386.

[14] Hanh, N.T., Binh, H.T.T., Hoai, N.X. and Palaniswami, M.S. (2019) An Efficient Ge-

netic Algorithm for Maximizing Area Coverage in Wireless Sensor Networks. *Information Sciences*, **488**, 58-75. https://doi.org/10.1016/j.ins.2019.02.059

[15] Wasay, H.A. and Periyasamy, K. (2023) Estimation of Coverage and Energy in Bio Inspired Wireless Sensors Using Harris Hawk Algorithm. *Indonesian Journal of Electrical Engineering and Computer Science*, **30**, 1813-1820. https://doi.org/10.11591/ijeecs.v30.i3.pp1813-1820

[16] Chowdhury, A. and De, D. (2021) Energy-Efficient Coverage Optimization in Wireless Sensor Networks Based on Voronoi-Glowworm Swarm Optimization-K-Means Algorithm. *Ad Hoc Networks*, **122**, Article ID: 102660. https://doi.org/10.1016/j.adhoc.2021.102660

[17] Trong-The Nguyen, T.N., Trong-The Nguyen, T.D., Thi-Kien Dao, T.N. and Trinh-Dong Nguyen, V.N. (2023) An Improved Honey Badger Algorithm for Coverage Optimization in Wireless Sensor Network. *Journal of Internet Technology*, **24**, 363-377. https://doi.org/10.53106/160792642023032402015

[18] Fu, W., Yang, Y., Hong, G. and Hou, J. (2021) WSN Deployment Strategy for Real 3D Terrain Coverage Based on Greedy Algorithm with DEM Probability Coverage Model. *Electronics*, **10**, Article No. 2028. https://doi.org/10.3390/electronics10162028

[19] Tuba, E., Tuba, M. and Beko, M. (2017) Mobile Wireless Sensor Networks Coverage Maximization by Firefly Algorithm. 2017 27*th International Conference Radioelektronika* (*RADIOELEKTRONIKA*), Brno, 19-20 April 2017, 1-5. https://doi.org/10.1109/radioelek.2017.7937592

[20] Dehghani, M., Montazeri, Z., Trojovská, E. and Trojovský, P. (2023) Coati Optimization Algorithm: A New Bio-Inspired Metaheuristic Algorithm for Solving Optimization Problems. *Knowledge-Based Systems*, **259**, Article ID: 110011. https://doi.org/10.1016/j.knosys.2022.110011