Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.076965

ARTICLE

Amplitude-Ensemble Quantum-Inspired Tabu Search Algorithm for Wireless Sensor Network Deployment

Kuo-Chun Tseng*, I-Chia Chen and Yu-Chieh Cho

Department of Computer Science and Information Engineering, National Ilan University, Yilan City, Taiwan

*Corresponding Author: Kuo-Chun Tseng. Email: kctseng@niu.edu.tw

Received: 29 November 2025; Accepted: 15 June 2026; Published: 23 July 2026

ABSTRACT: Wireless Sensor Networks (WSNs) are important infrastructure for smart-city applications, such as environmental monitoring, public safety, and smart transportation. However, finding effective sensor locations is an NP-hard problem because a deployment must satisfy sensing coverage and communication connectivity while minimizing the number of sensors. Following the basic framework of a previous study, this study replaces the original optimization algorithm with the Amplitude-Ensemble Quantum-inspired Tabu Search (AEQTS) algorithm and retains the same entanglement-like initialization strategy, resulting in the proposed AEQTSwE (AEQTS with Entanglement) framework for the WSN deployment problem. AEQTSwE uses a quantum-inspired search mechanism and an ensemble update strategy to explore the solution space more efficiently, while the retained initialization strategy provides high-quality initial deployments. Experimental results show that AEQTSwE reduces the number of deployed sensors while satisfying the required coverage and connectivity constraints. It also converges faster and produces more stable solutions than existing approaches under different conditions. Sensitivity, ablation, statistical, and complexity analyses further show that AEQTSwE has low parameter sensitivity, stable performance, and potential for larger and more complex deployment scenarios.

KEYWORDS: Wireless sensor network deployment; sensor deployment; quantum-inspired optimization; smart-city applications

1 Introduction

With the rapid development of Artificial Intelligence (AI) and the Internet of Things (IoT), the idea of smart cities is gradually moving from vision to reality. In a smart-city framework, AI plays a key role in real-time decision-making and automated control, and its performance depends heavily on timely data collection and accurate sensing. To achieve this, sensor deployment has become an essential part of smart-city infrastructure [1,2].

However, real-world smart-city environments are complex. Deployment may take place in two-dimensional (2D) or even three-dimensional (3D) spaces, and different sensors can have different constraints and specifications in practical scenarios. Communication performance and compatibility among sensors are also important [3]. These factors further increase the difficulty of the Sensor Placement Problem (SPP), which is known to be a challenging combinatorial optimization problem. Several variants of the SPP have been proven to be NP-complete or NP-hard [4,5]. Therefore, approximation and metaheuristic optimization methods are often required to obtain good solutions for large-scale instances.

Copyright © 2026 The Authors. Published by Tech Science Press. This work is licensed under a Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In most practical settings, sensors are required to have basic networking capabilities so that they can communicate and forward data to a sink node. For this reason, many studies focus on the deployment of Wireless Sensor Networks (WSNs). WSNs can be used in many applications such as traffic monitoring, environmental sensing, infrastructure safety, and energy management [1,6]. However, designing a WSN is not simply placing many sensors in random positions. It involves several conflicting objectives, including maximizing coverage, ensuring network connectivity, improving data transmission efficiency, reducing energy consumption, and minimizing deployment cost [6,7]. These conflicting goals make WSN deployment even more difficult. Nevertheless, solving this problem effectively can provide an important theoretical basis for future smart-city infrastructure and can help improve the accuracy of AI-based decisions.

To focus on the effects of coverage and connectivity, this study first considers a 2D deployment scenario with homogeneous sensors, where all sensors share the same sensing radius and communication range. Although simplified, this setting still reflects the core characteristics of many practical applications, such as indoor monitoring or planar-area deployment. It also provides a solid foundation for future extensions to heterogeneous sensors and 3D terrains. Based on this setting, we formulate the problem as a discrete combinatorial optimization task that aims to minimize the number of required sensors while satisfying multiple deployment constraints.

Although many metaheuristic methods are used to solve this NP-hard deployment problem, they still have major limitations in practice. Both traditional and hybrid algorithms often suffer from high parameter sensitivity, complex designs, and premature convergence. To address these limitations, this study develops a structurally simple optimization framework by replacing the core algorithm in the framework proposed by Kuo et al. [7]. Specifically, we adopt a quantum-inspired search mechanism proposed in our previous work [8]. This mechanism not only accelerates convergence and improves solution quality but also reduces parameter dependence and maintains structural simplicity.

Contributions

To obtain an efficient approximate solution for this problem, we adopt the quantum-inspired algorithm proposed in our earlier work [8] and evaluate it under standard WSN scenarios. The results show that it reduces the number of sensors while keeping sensing accuracy and connectivity stable, indicating its potential for large-scale automated WSN deployment in future smart cities. The main contributions of this study are as follows:

1. We propose the Amplitude-Ensemble Quantum-inspired Tabu Search algorithm (AEQTS) with Entanglement initialization (AEQTSwE), a quantum-inspired deployment algorithm that integrates the amplitude-based update mechanism of AEQTS [8] into the Quantum-inspired Tabu Search algorithm with Entanglement initialization (QTSwE) framework [7]. In addition, a historical global-best set is used to guide the search. The proposed method preserves a simple, low-parameter structure while enhancing stability, convergence speed, and solution quality under various WSN constraints, with reduced reliance on explicit balancing between exploration and exploitation.
2. This study also demonstrates that the historical global-best update strategy outperforms the original current-best update strategy used in QTSwE [7]. Experimental results show that, across the eight Uniform and Non-Uniform benchmark settings, AEQTSwE reduces the mean sensor count by about 10.6% compared to QTSwE-CB [7] (our implemented version of QTSwE [7]). In detailed data comparisons, the median sensor count is reduced by about 9.6%, while the standard deviation is reduced by about 61.5%.
3. This work provides a comprehensive evaluation of AEQTSwE through parameter sensitivity analysis, ablation studies, statistical significance tests, and computational complexity analysis. These analyses show that the proposed method has low parameter sensitivity, that its main optimization ability comes

from the quantum-inspired algorithm update mechanism, and that its main scalability bottleneck lies in the repair and evaluation procedures. The framework can also be extended to heterogeneous sensor deployment, dynamic re-positioning, 3D terrain deployment, and more complex real-world geographical maps containing physical obstacles and dynamic constraints [9,10]. This study offers a scalable foundation for future studies.

2 Related Work

The deployment and coverage optimization of WSNs has long been regarded as one of the most important and challenging research topics in this field, and it is also a key foundation for future AI and smart-city systems. The problem is essentially a high-dimensional combinatorial optimization task, where multiple objectives–such as coverage, network connectivity, sensing quality, and network lifetime–must be satisfied under limited cost and energy budgets.

Early studies modeled the coverage and deployment problem [11,12], and later work refined the probabilistic sensing model [13]. Several surveys also summarized WSN architectures, coverage models, deployment strategies, and related challenges [14–18], showing that deployment and coverage optimization remains a central and unresolved issue. A recent review by Egwuche et al. [19] further confirms that, even with modern machine learning techniques, significant research gaps still persist in WSN coverage and deployment.

Based on the existing literature, WSN deployment methods can be roughly divided into three categories: (a) heuristic and deterministic methods, (b) machine-learning-based methods, and (c) metaheuristic methods. In this work, we also highlight the development of quantum-inspired algorithms within this context, which will be explained in the following sections.

2.1 Heuristic and Deterministic Methods

Early work by Dhillon and Chakrabarty [20] introduced Max-Avg-Cov and Max-Min-Cov, which maximize average coverage and minimum coverage. Both methods outperform random and uniform deployment. Zou and Chakrabarty [11,12] proposed a probabilistic sensing model with confidence levels and developed Max-Miss and Min-Miss to control missed-detection probability, achieving more reliable coverage. Zhang et al. [4] formulated the deployment problem as an NP-hard integer linear program and proposed Diff-Deploy, which improves computational efficiency through the linear shift-invariant property and yields better coverage than Min-Miss and random deployment. Other studies used analytical or deterministic deployment based on geometric grids, analytical models, or dynamic programming. When the sensing radius is fixed and the area boundary is known, regular layouts such as square, triangular, or hexagonal grids can achieve full coverage and good connectivity with few sensors [14].

On the other hand, some studies [13,21,22] studied sensor deployment from the viewpoint of uncertain sensing. They proposed the Evidence-Based Deployment Algorithm (EBDA) and Efficient Evidence-Based sensor Deployment Algorithm (E2BDA), which use the transferable belief model for cooperative target detection while considering both coverage and connectivity. This uncertainty-based sensing model later became one of the main models used in follow-up research. Senouci et al. [22] also tested these methods on the ArduiNet experimental platform to show that they work in real environments. Li et al. [23] further modeled the deployment problem as a dynamic programming problem and used a step-by-step decision process to place sensors while balancing coverage and cost.

These methods offer high interpretability and well-defined models, making them strong references for regular sensing environments. However, their computational cost grows rapidly in large, irregular, or

constraint-rich environments. As a result, deterministic and traditional heuristic approaches serve mainly as theoretical baselines or small-scale deployment solutions and are difficult to extend to large-scale or multi-objective scenarios.

2.2 Machine-Learning-Based Methods

With the progress of artificial intelligence, Machine Learning (ML) has been widely used in many WSN functions, such as energy-aware routing, traffic prediction, anomaly detection, and k-barrier prediction [15,19]. In coverage and deployment problems, some studies use Reinforcement Learning (RL) to design sleep scheduling or node on/off strategies. These methods learn how to balance remaining energy and coverage quality to achieve area coverage and energy control. For example, Egwuche et al. [19] also summarized several works that use ML or RL to improve coverage or network lifetime.

However, survey papers consistently point out that the main role of machine learning in WSNs is still routing, data processing, and behavior prediction, rather than directly solving node deployment or coverage optimization [16,19]. Deployment is a one-time decision problem and lacks large labeled datasets for training. In addition, constraints such as coverage, connectivity, and cost are hard to include in an end-to-end ML model. Because of this, ML cannot easily replace optimization algorithms for deployment design. Therefore, ML is currently used more as a supporting tool, such as predicting environmental conditions or estimating coverage quality, rather than being the main optimization engine for coverage and deployment problems.

2.3 Metaheuristic Methods

In contrast, metaheuristic methods based on nature-inspired and evolutionary computation have become the mainstream solutions for deployment and coverage optimization. Many survey papers report that for multi-objective problems such as coverage, connectivity, and energy consumption, researchers commonly use Particle Swarm Optimization (PSO) [24], Genetic Algorithms (GA) [25], Differential Evolution (DE) [26], Grey Wolf Optimizer (GWO) [27], Artificial Bee Colony (ABC) [28], Ant Colony Optimization (ACO) [29], Harmony Search Algorithm (HSA) [30], and various multi-objective evolutionary algorithms [31]. For example, Wang et al. [32] proposed an improved GWO for node coverage optimization, using a nonlinear convergence factor, elite strategy, and dynamic mutation strategy to increase population diversity and escape local optima. ZainEldin et al. [25] introduced a GA-based dynamic deployment technique with variable-length encoding and a new two-point crossover operator, targeting maximum coverage with minimum node count. Gorkemli and Al-Dulaimi [28] studied the performance of fast ABC in dynamic deployment scenarios. Al-Fuhaidi et al. [30] used HSA to design a high-coverage deployment model for heterogeneous WSNs. Recent advances in 2025 have further optimized complex network deployments. For example, Priyadarshi [10] proposed a customized PSO framework that utilizes intelligent role assignment to effectively balance coverage quality and energy consumption in heterogeneous WSNs. Furthermore, to overcome the limitations of single metaheuristics, Anka [9] introduced a Hybrid Genetic Particle Swarm Optimization (HGPSO) method that integrates the global exploration of GA with the local exploitation of PSO to simultaneously optimize coverage, connectivity, redundancy, and energy efficiency. Overall, these works show that metaheuristics can handle multiple objectives and complex constraints with reasonable computational cost, making them currently the most effective and flexible solution family in practice.

The rapid evolution and necessity of these advanced techniques have been thoroughly documented in recent high-quality survey studies. Dokeroglu et al. [33] reviewed over 150 new metaheuristics to identify the 23 most influential algorithms, emphasizing that avoiding premature convergence in complex optimization requires hybrid, adaptive, and machine-learning-integrated mechanisms. More specifically in the domain of WSN, Houssein et al. [34] provided a comprehensive review confirming that tackling complex WSN

challenges, such as multi-objective deployment and energy efficiency, increasingly requires moving beyond traditional single algorithms to more sophisticated metaheuristic designs.

However, while these sophisticated and hybrid approaches improve optimization performance, several survey papers [2,16,19] also point out that most metaheuristics still share some common weaknesses. These include high sensitivity to parameter settings, limited portability due to problem-specific operator design, and relatively weak theoretical analysis and convergence guarantees. Therefore, an important research direction is how to keep their strong search ability while reducing parameter dependence and improving structural simplicity and analyzability.

Recent studies in the field of metaheuristics have also gradually explored natural or mathematical principles beyond animal behaviors to inspire algorithm design. This direction may help simplify the difficulty of balancing exploration and exploitation, reduce parameter dependence, and make algorithm implementation simpler. Such designs are also useful for dealing with real-world engineering challenges that are difficult to model. For example, Hussein et al. [35] proposed an algorithm inspired by quantum properties. This algorithm provides a structured way to balance exploration and exploitation, and its effectiveness was further tested on real-world engineering challenges. Later, Ibrahim et al. [36] designed an algorithm from the perspective of mathematical mechanisms by introducing the concept of slope into population-based search, thereby simplifying the overall algorithm design and reducing the number of control parameters. These recent trends are also consistent with the motivation of quantum-inspired metaheuristics, which aim to use more structured search representations and update mechanisms to maintain global search ability while reducing the dependence on manual parameter tuning and problem-specific characteristics.

2.4 Quantum-Inspired Metaheuristics

Quantum-inspired algorithms have been introduced as enhanced metaheuristics that represent solution probabilities with quantum bit (qubit) amplitudes and explore the search space through quantum measurements and rotation updates. Kuo et al. [7] proposed Quantum-inspired Tabu Search algorithm with Entanglement initialization (QTSwE), which integrates quantum-inspired tabu search with entanglement to generate dependency-aware initial solutions and then applies global and local search. Their results show that QTSwE satisfies coverage and connectivity requirements using fewer sensors and outperforms several existing deployment methods [37].

Overall, prior work reveals a clear progression. Heuristic and deterministic methods serve as baselines but scale poorly in large or complex settings. Machine learning is widely applied in WSNs but remains a supporting tool rather than a primary optimizer for deployment and coverage. Metaheuristics provide strong practical performance, and quantum-inspired methods further enhance search ability. Following this direction, this study adopts AEQTS [8] as the optimization engine, aiming for a simple and low-parameter framework that achieves faster convergence and better solution quality in WSN deployment.

The design concept of AEQTS [8] is inspired by Grover's search algorithm [38]. Grover's search algorithm is a well-known quantum algorithm that can find a target item in an unsorted database with a complexity of $O(\sqrt{N_{\mathrm{db}}})$. It mainly uses $O(\sqrt{N_{\mathrm{db}}})$ oracle queries and amplitude adjustments to maximize the amplitude, or probability, of the target solution. Grover's search algorithm [38] is simple, efficient, parameter-free, and easy to analyze, making it a useful reference model for designing quantum-inspired algorithms.

AEQTS [8] is inspired by the amplitude adjustment concept of Grover's search algorithm [38]. Before AEQTS [8], QTS [39] was the quantum-inspired algorithm whose overall structure was closest to Grover's search algorithm [38]. Therefore, the name AEQTS still follows the naming style of QTS. However, QTS [39] uses only the best and worst solutions in each iteration, which wastes much of the information contained

in the population. In contrast, AEQTS [8] successfully uses all population information. In each iteration, all solutions in the population are paired and used to update the corresponding quantum states one by one. This makes the convergence faster and more effective, without introducing any additional parameters.

Moreover, in our later studies, we observed some quantum-like effects in AEQTS [8]. Even for many problems where QTS [39,40] fails to converge, AEQTS [8] can still continue to converge well. Therefore, the performance and behavior of AEQTS [8] are substantially different from those of QTS [39]. AEQTS [8] should not be simply regarded as another extension or variant of QTS [39,40].

In addition, unlike traditional algorithms, AEQTS [8] does not need to explicitly balance exploration and exploitation. Therefore, it does not require separate operations for these two search behaviors, nor does it require extensive experiments to tune parameters caused by the trade-off between them. For these reasons, AEQTS [8] retains a close conceptual connection to quantum amplitude amplification while remaining simple and effective as a quantum-inspired metaheuristic.

3 Wireless Sensor Network Deployment Problem

This section defines the WSN deployment problem in three parts. First, the sensing characteristics of sensors and the related mathematical expressions are introduced. Then, the constraints of the deployment problem are described. Finally, the objective function and constraints of this problem are formulated.

3.1 Sensing Model

In this study, we consider a two-dimensional grid composed of $m \times n$ grid cells. This means the grid has a width of m and a height of n. Each cell g_{ij}, where $i \in \{1, 2, \ldots, m\}$ and $j \in \{1, 2, \ldots, n\}$, has its own event probability. A sensor may or may not be placed in each grid cell g_{ij}, and if a sensor is placed, only one sensor can be assigned to that cell.

The probability sensing model used for each g_{ij} in this study is similar to the models used in [2,7,13,21,37,41], as shown in Fig. 1. This model allows the deployment problem to be discretized more effectively, making it suitable for various optimization algorithms.

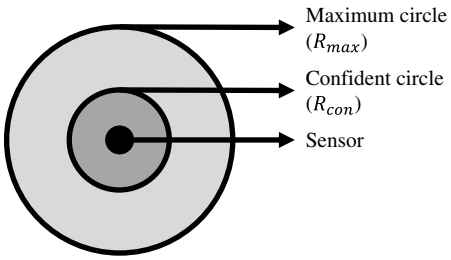


Figure 1: Sensing range model of a sensor.

In Fig. 1, the sensing range of a sensor is divided into a confident circle and a maximum circle. Events occurring inside the confident circle are detected with the highest probability. For example, if a sensor's maximum detection probability is 100%, then all events in the confident circle are detected with 100% probability. Events occurring between the confident circle and the maximum circle are detected with a decreasing probability, and all sensors follow a distance-based probability decay. These sensing rules are defined in Eq. (1), where $|sg_{ij}|$ represents the Euclidean distance between the sensor and the grid cell g_{ij} (as defined in Eq. (2)).

In this study, we assume $R_{con} = 1$ and $R_{max} = 5$, meaning that detection probabilities below 0.2 are considered undetectable. We also set $p_{\max} = 1$ and $\gamma = 1$ as commonly used values. Under this configuration, if a sensor is placed at the center of an 11 × 11 grid, then the detection probability for each g_{ij} can be computed using Eq. (1), as illustrated in Fig. 2.

$$P^s_{g_{ij}} = \begin{cases} p_{\max} & \text{if } \left\|sg_{ij}\right\| \le R_{con}, \\ \dfrac{1}{\left\|sg_{ij}\right\|^{\gamma}} & \text{if } R_{con} < \left\|sg_{ij}\right\| \le R_{max}, \quad p_{\max} \le 1,\ \gamma > 0,\ R_{con} = 1,\ R_{max} = 5 \\ 0 & \text{if } \left\|sg_{ij}\right\| > R_{max}. \end{cases} \tag{1}$$

$$d(s, g_{i'j'}) = \sqrt{(s_i - g_{i'})^2 + (s_j - g_{j'})^2} \tag{2}$$

$$P_{ij} = 1 - \prod_{s \in \eta} \left(1 - P^s_{ij}\right), \forall i, j \tag{3}$$

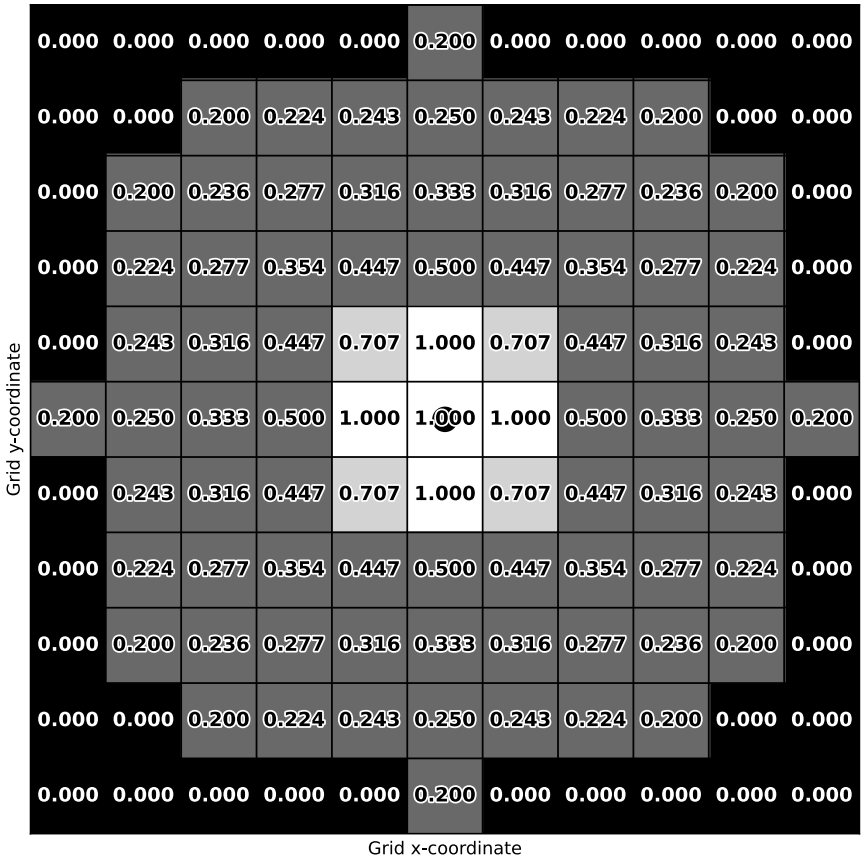


Figure 2: Sensor detection probability map. Darker colors represent lower detection probabilities, while brighter colors represent higher detection probabilities.

In addition, we must consider the detection probability when the sensing ranges of multiple sensors overlap. As shown in Eq. (3), let η be the set of all sensors. If several sensors cover the same grid cell g_{ij}, then the detection probability P_{ij} is calculated as one minus the product of the non-detection probabilities of all sensors. This represents the probability that at least one of the covering sensors can detect the event at g_{ij}.

For example, in Fig. 3, two sensors are placed at coordinates (5, 6) and (11, 6) (coordinates counted from the top-left corner starting from 1). For the grid cell at (7, 6), its detection probability is $P_{(7,6)} = 1 - ((1 - 0.5) * (1 - 0.25)) = 0.625$, where the individual detection probabilities follow Eq. (1) and the single-sensor distribution shown in Fig. 2.

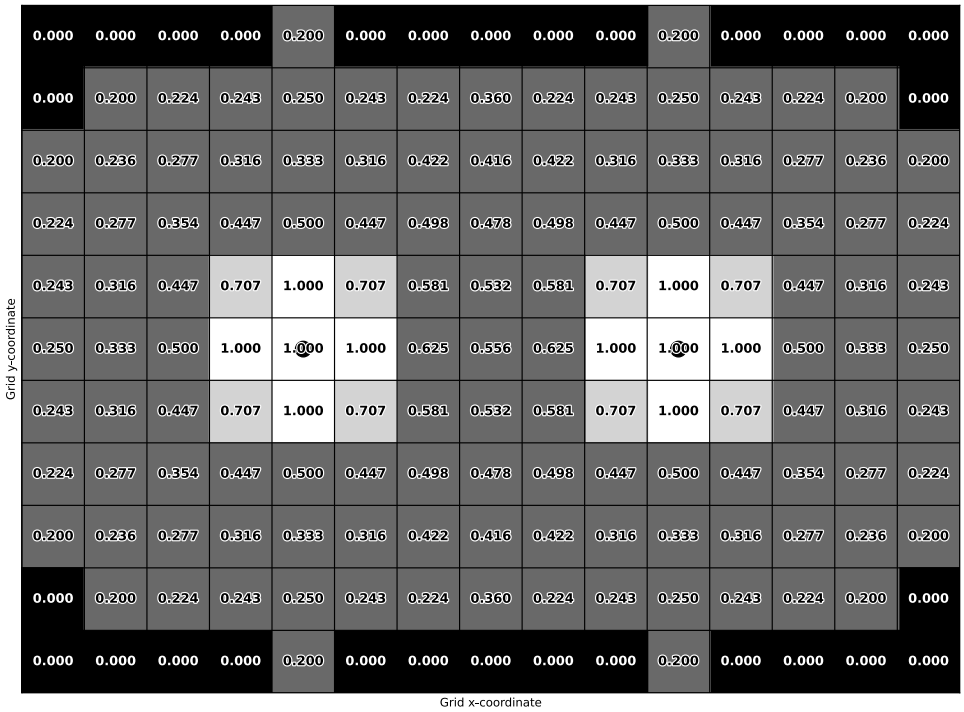


Figure 3: Grid-based detection probability with multiple sensors. Darker colors represent lower detection probabilities, while brighter colors represent higher detection probabilities.

Finally, we define a threshold value T_{ij} for each grid cell. Each cell g_{ij} in a scenario may have a different T_{ij} to represent different sensing hotspots. In common benchmark settings, a uniform distribution is usually used, where all T_{ij} values are set to the same number. This allows us to evaluate and compare the performance of different deployment algorithms.

3.2 Constraints in the Deployment Problem

After defining the deployment scenario, we then discuss the constraints of the deployment problem. This means that not every sensor deployment is a feasible solution. A deployment is feasible only if it satisfies the following two constraints:

1. All detection thresholds T_{ij} must be satisfied, as required by Eq. (4).
2. All sensors must be connected, meaning that every sensor must have a path to send data back to the sink. This must satisfy Eq. (5), where $\left|CC(\mathcal{G})\right|$ represents the number of connected components in the grid $\mathcal{G}$. The entire network must have exactly one connected component, i.e., $\left|CC(\mathcal{G})\right| = 1$. If any part of the network becomes disconnected, then $\left|CC(\mathcal{G})\right| \geq 2$.

$$P_{ij} \geq T_{ij}, \quad \forall i, j \tag{4}$$

$$\left|CC(\mathcal{G})\right| = 1 \tag{5}$$

3.3 Deployment Problem Formulation

In this study, the deployment configuration is represented by a binary matrix $D \in \{0,1\}^{m\times n}$, where $D_{ij} = 1$ indicates that a sensor is placed at grid cell g_{ij}, and $D_{ij} = 0$ otherwise. The matrix can be equivalently expressed as a binary vector $x \in \{0,1\}^{mn}$ to match the classical discrete optimization form.

Objective function: The goal of the deployment problem is to minimize the total number of deployed sensors as Eq. (6).

Coverage constraint: For each grid cell, the aggregated detection probability must satisfy the required threshold T_{ij} as Eq. (7), where $P_{ij}(x)$ is computed using the probabilistic sensing model defined in Eqs. (1)–(3).

Connectivity constraint: All deployed sensors must form a single connected component under the communication radius R_{comm} as Eq. (8), where $CC(\mathcal{G}(x))$ denotes the connected components of the induced communication graph $\mathcal{G}(x)$.

Final formulation: The deployment problem is summarized in Eq. (9).

$$\min f(x) = \sum_{i=1}^{m}\sum_{j=1}^{n} D_{ij} \tag{6}$$

$$g_1(x) = T_{ij} - P_{ij}(x) \le 0, \qquad \forall i, j \tag{7}$$

$$g_2(x) = |CC(\mathcal{G}(x))| - 1 = 0, \tag{8}$$

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & g_1(x) \le 0, \\ & g_2(x) = 0, \\ & x \in \{0,1\}^{mn} \end{aligned} \tag{9}$$

Since the formulation involves binary decision variables, nonlinear probabilistic coverage constraints, and graph-connectivity constraints, the resulting optimization problem is a nonlinear integer program and is NP-complete or NP-hard [4,5]. Therefore, exact optimization techniques become infeasible for realistic grid sizes, which motivates the use of our AEQTS [8] as the optimization engine.

4 AEQTS with Entanglement Initialization (AEQTSwE) for WSN Deployment

This section first introduces the AEQTSwE algorithm for the WSN deployment problem and explains its core design concept in detail. Then, it describes why the update strategy of AEQTSwE used in this work is different from that of the original AEQTS [8], based on the characteristics of this problem. Finally, the overall computational complexity and scalability of the proposed method are analyzed.

4.1 Proposed AEQTSwE Algorithm

This study adopts the AEQTS algorithm proposed by our team [8] as the core method for sensor deployment. Following the idea of Kuo et al. [7], we adapt AEQTS [8] to the deployment setting and obtain AEQTSwE, which handles both coverage and connectivity. The overall procedure is shown in Algorithm 1.

Compared with QTS [39], AEQTS [8] uses the information from the whole population. Without introducing any additional parameters, it achieves faster convergence and better search performance in

the tested settings. Another advantage of AEQTS [8] is that it reduces the need for separately designed exploration and exploitation operators. Traditional algorithms often introduce many parameters to balance these two search behaviors, which makes the algorithms more complex and requires more time for parameter tuning. In contrast, AEQTS [8] has a simple structure with very few parameters, and its performance is less sensitive to parameter settings. Once implemented, it can provide competitive performance with relatively low tuning effort.

Algorithm 1: AEQTSwE sensor deployment optimization process

1: $t \leftarrow 0$
2: Generate an initial solution s by entanglement-based preprocessing
3: Initialize the matrix of quantum bits $Q(t)$ by s
4: **while** $t <$ MaxIter **do**
5: $t \leftarrow t+1$
6: Measure $Q(t-1)$ multiple times to generate the neighborhood set N
7: **for** each $s \in N$ **do**
8: Repair s and evaluate $f(s)$
9: **end for**
10: If the best in N is better than the global-best, update it
11: Detect whether the process is stuck in a local optimum
12: **if** stuck **then**
13: Perform local search
14: **end if**
15: Select historical global-best set S^b (distinct, $|S^b| = |N|/2$) and the current-worst set S^w ($|S^w| = |N|/2$) from N
16: Update $Q(t)$ using S^b and S^w
17: **end while**

In Algorithm 1, the procedure is similar to the method of Kuo et al. [7] with two main changes. AEQTSwE updates angles using all population solutions for pairing, and it forms pairs between the historical global-best set and the current-worst set. The main components of Algorithm 1 are explained below.

Line 2: In iteration 0, the initial solution is generated using an entanglement mechanism. Because WSN deployment has certain characteristics–placing sensors too close causes overlap and wastes resources, while placing them too far apart may break connectivity–the distance between sensors must satisfy a minimum spacing. Similar to quantum entanglement, we derive an effective distance $d_{eff} = \min(R_{max}, T_g^{-1/\gamma})$ based on T_g, R_{con}, and R_{max}, and use it as the minimum allowable distance between any two sensors. A valid mask M is then constructed (Algorithm 2) to prevent sensors from being placed near the boundaries. After that, initial sensor positions are randomly placed within the valid region M, and every pair of selected positions must be at least d_{eff} apart to form s.

Line 3: Initialize the qubit matrix $Q(0)$ using s. The matrix $Q(0)$ has the same dimension as the deployment grid, defined as $Q(0) = \{q_{ij} \mid i = 1, \ldots, m,\ j = 1, \ldots, n\}$, where each q_{ij} is a qubit. At the beginning, all qubits are set to the quantum state $\alpha = \beta = 1/\sqrt{2}$. Then, we map each q_{ij} to the corresponding value in s_{ij}; if $s_{ij} = 1$, the corresponding qubit is forced to $\alpha = 0$ and $\beta = 1$ (i.e., mapped to value 1), while all other qubits remain unchanged.

Line 6: Measure the qubit matrix $Q(t-1)$ to generate the neighborhood solution set N. Each solution s is produced by generating a random number at every position in $Q(t-1)$ and comparing it with the corresponding $|\beta|^2$ value; if the random number is greater, the bit is set to 0, otherwise it is set to 1. Each solution therefore represents a possible deployment.

Lines 7–9: Repair each generated solution using Algorithm 3. The process first attempts to remove sensors one by one; if removing a sensor does not degrade either coverage or connectivity, it is removed. If coverage becomes insufficient, a sensor is added at the grid cell with the lowest coverage until coverage satisfies Eq. (4). If connectivity is not satisfied, the algorithm identifies the closest disconnected components, finds the sensor pair with the shortest distance, and places additional sensors along that path. This continues until all components become connected, satisfying Eq. (5). After repair, each solution s is evaluated, and its fitness $f(s)$ is defined as the number of sensors used (assuming that the repair mechanism can always guarantee 100% connectivity and coverage rate). In fact, when the historical global-best update strategy is used on smaller problems, the repair mechanism does not show a very obvious advantage. Even without the repair mechanism, the algorithm can still achieve an average coverage rate close to 100% and full connectivity. Only when the problem size becomes larger does the repair mechanism help the algorithm continue to converge. A more detailed discussion and analysis are provided in Section 5.3.2.

Line 11: If the global-best is not updated for a number of iterations equal to the variable *detection_iteration*, the algorithm concludes that it has fallen into a local optimum.

Lines 12–14: After entering the local search phase, the algorithm randomly selects a fixed number of solutions (local size), determined by a given parameter, and tests them by attempting to move each one to one of the eight surrounding grid cells. A movement is accepted only if both coverage and connectivity remain unchanged; otherwise, the sensor returns to its previous position and the local search for that sensor stops. If the new position overlaps with another sensor, the two sensors are merged. If the sensor reaches the final tested position and all movements still maintain the same coverage and connectivity, the last movement is kept to increase solution diversity.

Line 15: Select $|N|/2$ solutions from the historical global-best solution set to form S^b, ensuring that no duplicate solutions are chosen (this set can also be maintained across iterations). Similarly, select solutions from the current-worst solution set in N to form S^w. This is the main difference between our approach and QTSwE [7]. Furthermore, it significantly improves the algorithm's performance by 10.6% and makes it much more stable. WSN deployment problem has a large combinatorial search space, i.e., $2^{m\times n}$, and its fitness function considers several factors, such as connectivity and the number of sensors. Therefore, the solutions in different iterations may vary greatly. If the original pairing strategy of AEQTS [8] is used, the quantum states may oscillate repeatedly. As a result, the convergence direction becomes less clear, the convergence speed becomes slower, and the algorithm may have difficulty converging to better solutions. Therefore, in this study, we use the historical global-best solution set for pairing, so that the overall algorithm can converge in a clearer direction. However, for problems with less severe oscillation, the original pairing strategy may still be better because it allows the algorithm to perform a more detailed search. This design keeps the modification relatively simple. A more detailed discussion and explanation of the oscillation phenomenon and the historical global-best pairing update strategy are provided in Section 4.2.

Line 16: Sort S^b and S^w according to solution quality to obtain two ordered sequences, $S^b = (S_1^b, S_2^b, \ldots, S_{|N|/2}^b)$ and $S^w = (S_1^w, S_2^w, \ldots, S_{|N|/2}^w)$, respectively. Then, (S_k^b, S_k^w) denotes the k-th pair, where $k = 1, 2, \ldots, |N|/2$. Each solution in the pair (i, j) denotes the position of the corresponding qubit q_{ij} in the $m \times n$ qubit matrix. For each pair, the rotation angle at position (i, j) is defined as $\Delta\theta_{ij}/k$, so that higher-ranked pairs contribute a larger rotation. In the k-th pair, if the (i, j)-th qubit satisfies $S_{k,ij}^b \neq S_{k,ij}^w$, the corresponding qubit q_{ij} is updated using the rotation matrix $R(\pm\Delta\theta_{ij})$ so that its probability distribution moves toward the state represented by $S_{k,ij}^b$. Through the repair mechanism and this update procedure, the algorithm can modify qubits whose amplitudes were forced to $\beta = 1$ during initialization, allowing all qubit states to be properly corrected.

Algorithm 2: Valid sensor mask generation

Compute the effective minimum distance $d_{eff} = min(R_{\max}, T_g^{-1/\gamma})$ from T_g, R_{con}, and $R_{\max}$
Initialize M as the feasible deployment region
for each candidate position i in the deployment area **do**
    Compute the minimum distance $d_{\text{edge}}(i)$ from i to the boundary
    if $d_{\text{edge}}(i) \geq d_{\text{eff}}$ **then**
        Mark i as valid in M
    else
        Mark i as invalid in M
    end if
end for
return M

Algorithm 3: Solution repair procedure

Compute the detection probability map p of the current deployment s
while there exists a removable sensor in s that keeps coverage and connectivity above the thresholds **do**
    Remove the selected sensor from s
    Update p for the new deployment
end while
while there exists an uncovered cell with $p < T_g$ **do**
    Find the cell with the minimum detection probability
    Place a sensor at this cell and update s
    Update p for the new deployment
end while
while s is not connected under R_{comm} **do**
    Identify two disconnected components with the shortest inter-component distance
    Find the closest pair of sensors u and v belonging to these components
    Place additional sensor(s) along the shortest path between u and v respecting R_{comm} and $R_{\max}$
    Update the connectivity of s
end while
return s

4.2 Suitability of Historical Global-Best Pairing for WSN Deployment

In the original update strategy of AEQTS [8], the solution set in each iteration is divided into a current-best solution set and a current-worst solution set, denoted by S^b and S^w, respectively. In this subsection, S^b and S^w are used as general symbols for the two solution sets involved in the pairing process. Their actual construction can be determined by the corresponding selection rule. After sorting, the solutions in these two sets are paired and updated by comparing $S^b_{k,ij}$ with $S^w_{k,ij}$, where k denotes the k-th pair and (i, j) denotes the position of a qubit in the $m \times n$ qubit matrix. Each qubit represents the probability of placing a sensor in a specific grid cell. Let $p^{(t)}_{ij}$ be the probability that the qubit q_{ij} is measured as '1' in the t-th iteration. This probability can be represented by an angular state as $p^{(t)}_{ij} = \sin^2 \phi^{(t)}_{ij}$. Thus, updating the qubit is equivalent to updating the angle $\phi^{(t)}_{ij}$.

In each iteration, the qubit is rotated according to the difference between $S^b_{k,ij}$ and $S^w_{k,ij}$ in each pair. For the qubit q_{ij}, if $S^b_{k,ij} = 1$ and $S^w_{k,ij} = 0$, it means that placing a sensor in this grid cell is more favorable. Therefore, the qubit state is pushed toward '1'. Conversely, if $S^b_{k,ij} = 0$ and $S^w_{k,ij} = 1$, the qubit state is pushed toward 0. If $S^b_{k,ij} = S^w_{k,ij}$, the angle is not updated. Therefore, the update direction can be simplified as $d^{(t)}_{k,ij} \in \{-1, 0, +1\}$, where $d^{(t)}_{k,ij} = +1$ means updating toward '1', $d^{(t)}_{k,ij} = -1$ means updating toward '0', and $d^{(t)}_{k,ij} = 0$ means no update.

Then, the update in iteration $t + 1$ can be simplified as

$$\phi^{(t+1)}_{ij} = \phi^{(t)}_{ij} + \lambda^{(t)}_{k,ij} d^{(t)}_{k,ij},$$

where $\lambda^{(t)}_{k,ij}$ denotes the update magnitude by the k-th pair to the qubit q_{ij}. After multiple iterations, the angle of this qubit becomes

$$\phi^{(T)}_{ij} = \phi^{(0)}_{ij} + \sum_{t=0}^{T-1} \sum_{k=1}^{|N|/2} \lambda^{(t)}_{k,ij} d^{(t)}_{k,ij}.$$

This means that if the update direction of this qubit is generally consistent across pairs and iterations, $\phi^{(0)}_{ij}$ can be effectively changed. In contrast, if the update direction changes frequently, positive and negative updates may cancel each other out. As a result, $\sum_{t=0}^{T-1} \sum_{k=1}^{|N|/2} \lambda^{(t)}_{k,ij} d^{(t)}_{k,ij}$ becomes relatively small, and the angle may oscillate around the middle region. Therefore, the key factor for convergence is not only the magnitude of the rotation angle, but also whether the update direction $d^{(t)}_{k,ij}$ remains consistent across iterations.

In the WSN deployment problem, when coverage and connectivity are already satisfied or nearly satisfied, many solutions may have the same or similar number of sensors but clearly different spatial structures. These solutions form a large near-equivalent solution set. In this case, if the current-best update mechanism is used, S^b is reselected from the current population in each iteration, which is similar to repeatedly sampling from this near-equivalent set. As a result, the update direction of the qubit q_{ij} may change frequently across iterations, and its expected accumulated update tends to be close to zero, that is, $\mathbb{E}[\sum_{k=1}^{|N|/2} \lambda^{(t)}_{k,ij} d^{(t)}_{k,ij}] \approx 0$. Then, $\sum_{k=1}^{|N|/2} \lambda^{(t)}_{k,ij} d^{(t)}_{k,ij}$ becomes a zero-drift accumulation and cannot effectively change ϕ_{ij}. This is the main reason for oscillation.

In contrast, when the historical global-best update strategy is used, the update direction can remain more stable across iterations. In this case, $d^{(t)}_{k,ij}$ tends to be more consistent, which means that the expected accumulated update is not close to zero and can remain stable. Therefore, the accumulated update can

continue in a similar direction and effectively drive ϕ_{ij} toward '0' or '1'. For this reason, the historical global-best pairing strategy can provide a clearer convergence direction for the WSN deployment problem. It also makes the changes in the overall sensor layout more fine-grained and moderate, so the search behavior becomes closer to local refinement.

However, it should be emphasized that this does not mean that the historical global-best strategy is the best choice for all problems. If update-direction oscillation does not occur, the current-best pairing strategy may be more suitable, while the historical global-best pairing strategy may lead to premature convergence. This is also the only implementation-dependent part of AEQTS [8]. Therefore, different pairing strategies may need to be tested first, and the final choice should be made after the problem characteristics are better understood.

4.3 Computational Complexity and Scalability Analysis

To further explain the scalability of AEQTSwE on larger-scale problems, this subsection briefly analyzes its computational complexity. Let $L = m \times n$ denote the total number of grid cells in the deployment area. For simplicity, let $n_{\text{pop}} = |N|$ denote the population size in each iteration, G denote the maximum number of iterations (denoted as MaxIter), and n_s denote the number of deployed sensors in a single candidate solution, where $n_s \leq L$.

In each iteration, AEQTSwE first performs measurement based on the qubit matrix to generate n_{pop} candidate solutions. Since the length of each candidate solution is L, the time complexity of this step is $O(n_{\text{pop}}L)$. In addition, the qubit update between the historical global-best set and the current-worst set also needs to scan the candidate solution vectors and update the corresponding qubit amplitudes. Therefore, its time complexity is also $O(n_{\text{pop}}L)$.

The main computational cost comes from repair and fitness evaluation. For a single candidate solution, coverage evaluation needs to compute and update the detection probability of grid cells based on the currently deployed sensors. Its cost can be conservatively estimated as $O(n_sL)$. In addition, connectivity checking needs to examine the connection relations among deployed sensors. If it is analyzed using pairwise sensor relations, its cost can be expressed as $O({n_s}^2)$.

However, the repair procedure is not only a single evaluation. It may repeatedly try to remove redundant sensors, cover uncovered cells, and fix disconnected components. Therefore, let K_r denote the number of repair checks or repair operations required by the repair procedure for a single candidate solution. Specifically, K_r can be regarded as the sum of the following operations:

$$K_r = K_{\text{remove}} + K_{\text{cover}} + K_{\text{connect}},$$

where K_{remove} denotes the number of attempts to remove sensors, K_{cover} denotes the number of sensors added to satisfy coverage, and K_{connect} denotes the number of sensors added or checked to fix connectivity.

In each repair operation, in the most conservative case, coverage evaluation and connectivity checking may need to be performed again. Therefore, the cost of a single repair operation can be approximately expressed as $O(n_sL + {n_s}^2)$. Since a single candidate solution may require K_r repair operations, the repair and evaluation cost for one candidate solution can be expressed as $O(K_r(n_sL + {n_s}^2))$. Therefore, the cost of repairing and evaluating n_{pop} candidate solutions in each iteration is $O(n_{\text{pop}}K_r(n_sL + {n_s}^2))$.

By combining measurement, repair/evaluation, and qubit update, the overall time complexity of AEQTSwE in one iteration can be expressed as

$$O(n_{\text{pop}}L) + O(n_{\text{pop}}K_r(n_sL + {n_s}^2)) + O(n_{\text{pop}}L).$$

After ignoring constant factors, it can be simplified as $O(n_{\text{pop}}(L + K_r(n_sL + {n_s}^2)))$. By further multiplying it by the total number of iterations G, the overall time complexity becomes $O(Gn_{\text{pop}}(L + K_r(n_sL + {n_s}^2)))$.

Next, consider the worst-case complexity. Since the sensor count cannot exceed the number of grid cells, we have $n_s \leq L$. Therefore, $n_sL = O(L^2)$ and ${n_s}^2 = O(L^2)$, so $n_sL + {n_s}^2 = O(L^2)$. In addition, in the most conservative naive repair implementation, the repair procedure may require more repair checks or repair operations as the grid size increases. For example, the algorithm may need to try removing many sensors, or gradually add sensors to satisfy coverage and connectivity. In the worst case, the number of these repair operations can be upper-bounded by L. Therefore, we can denote $K_r = O(L)$. By substituting $K_r = O(L)$ and $n_sL + {n_s}^2 = O(L^2)$ into the overall complexity, we obtain $O(Gn_{\text{pop}}(L + L \times L^2)) = O(Gn_{\text{pop}}L^3)$.

This analysis shows that, when the population size n_{pop} and maximum iterations G are fixed, the computational cost of AEQTSwE grows polynomially with grid size. The bound $O(Gn_{\text{pop}}L^3)$ is a conservative worst-case upper bound, which mainly corresponds to the case where coverage and connectivity are directly recomputed during repeated repair operations. In practical execution, K_r is usually much smaller than L, and actual cost can be further reduced by optimized repair and evaluation implementations. For example, when the grid size increases from 50×50 to 100×100, L increases by four times. If K_r does not increase significantly, the main repair/evaluation cost may increase by about $4^2 = 16$ times. In the conservative worst case where $K_r = O(L)$, the cost may increase by about $4^3 = 64$ times. Therefore, the practical scalability bottleneck lies in the repeated repair and coverage/connectivity evaluation steps rather than in the AEQTSwE itself.

5 Experimental Results

This section describes the experimental environment and parameter settings, and evaluates the performance of the proposed AEQTSwE sensor deployment method. The proposed method is implemented in Python. Following the scenario design of Kuo et al. [7], the simulation environment is a 50×50 grid map that includes two types of scenarios: Uniform and Non-Uniform distributions. The proposed method is compared with several representative sensor deployment strategies, including Random-Uniform, Random-Bernoulli, Grid, Max-Min-Cov, Max-Avg-Cov, Min-Miss, Diff-Deploy, Differentiated Deployment Algorithm (DDA), Bernoulli Deployment Algorithm (BDA), Potential Field Deployment Algorithm (PFDA), QTSwE (results from Aitsaadi et al. [37] and Kuo et al. [7]), QTSwE-CB [7] (the version of QTSwE that we implemented uses the current-best (best-per-iteration) update strategy), QTSwE-GB (QTSwE using historical global-best update strategy (proposed in this work)), and AEQTSwE. Our QTSwE-CB implementation produces results close to those reported in [7]. Based on this implementation, we further developed the QTSwE-GB and AEQTSwE variants, both of which show better performance than QTSwE-CB [7].

The population size of the algorithm is set to 100, and the maximum number of iterations is set to 500. The initial value of the quantum probability amplitude $|\beta|^2$ is set to 0.5. The local search procedure is triggered when the best solution has not improved for 10 consecutive iterations (Detection iteration). In this stage, the algorithm randomly selects 10 solutions (Local size) for a local search procedure, generates 10 neighboring solutions, and replaces the original solution with the one that has better fitness. The rotation angle for updating the quantum states is set to $\Delta\theta = 0.1\pi$. The main algorithm parameters are summarized in Table 1. Other parameters are set as $R_{max} = 5$, $R_{\text{comm}} = 1.414$, $p_{\max} = 1$, and $\gamma = 1$. Four scenarios are then used for comparison, including $T_g = 0.3$, 0.6, 0.9, and a Non-Uniform distribution. For both the Uniform and Non-Uniform distributions, each experiment is independently executed 30 times, and the results are averaged. All averaged values are rounded. Convergence plots are also provided for each scenario, where transparent regions indicate the fluctuation range of the algorithm, and solid lines represent the averaged global-best value across iterations.

Table 1: Parameters of the AEQTSwE.

Parameter	Value
Population size ($\|N\|$)	100
Local search size	10
Max. iter. (MaxIter)	500
Rotation angle ($\Delta\theta$)	0.1π
Detect iterations	10

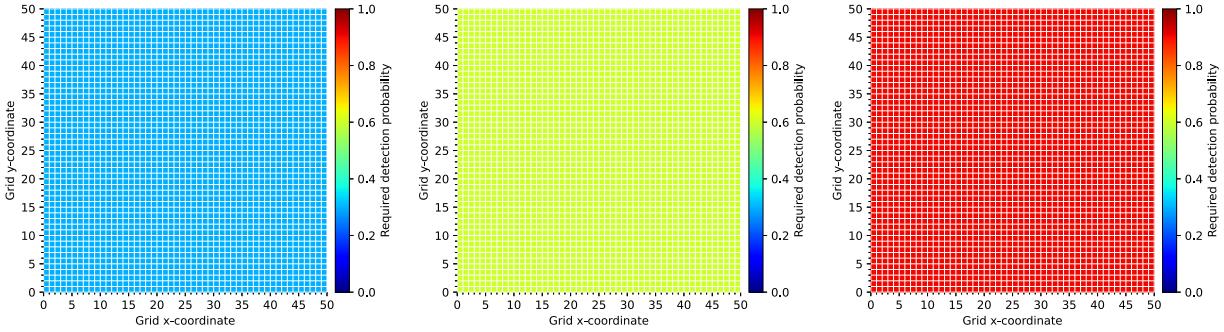
Note: "Max. iter." denotes the maximum number of iterations.

All methods are also evaluated under two conditions: (a) connectivity treated as a hard constraint, and (b) connectivity not treated as a constraint. When connectivity is a constraint, any sensor that cannot reach the sink triggers a repair process until all sensors eventually become connected. When connectivity is not a constraint, higher connectivity simply contributes to a better fitness score during evaluation (number of disconnected sensors $\times$ $(m \times n)$ + number of connected sensors + number of uncovered grids). The repair mechanism does not check or enforce connectivity in condition (b). These two evaluation settings allow a more comprehensive and reliable assessment of the sensor deployment quality.

5.1 Uniform Distribution

Fig. 4 shows three uniform-distribution scenarios with required detection probabilities of 0.3, 0.6, and 0.9, which correspond to low, medium, and high sensing density. From Tables 2–7, we can observe that the method of Kuo et al. [7] still performs better than other classical approaches. Our implementation, QTSwE-CB [7], produces results similar to those reported by Kuo et al. [7], except for Scenario 1 when connectivity is not considered. Overall, the original behavior of QTSwE is successfully reproduced. In addition, we find that using historical global-best solutions for updating can significantly improve the solution quality, and the AEQTSwE variant, which adopts this update strategy, achieves the best performance. The final deployment results for the three scenarios are shown in Fig. 5. We further compare QTSwE-CB [7], QTSwE-GB, and AEQTSwE using the convergence plots in Figs. 6 and 7. In all scenarios, AEQTSwE converges faster and reaches the best final result. This shows that its update strategy can more effectively use information from the entire population. Specifically, by using the historical global-best solution set, the update strategy provides a consistent update direction while retaining sufficient diversity. This approach accelerates convergence while maintaining high-quality final solutions. In addition, Table 8 shows more detailed results, such as the best, worst, mean, standard deviation, and median values. These values show that AEQTSwE not only performs better but is also more stable.

Both AEQTSwE and QTSwE-GB use the historical global-best solution to update qubits. They likely perform better because of the nature of this problem. If the algorithm only uses the current iteration's best and worst solutions for updates, the qubit probabilities cannot converge easily. This result also indicates that AEQTSwE can effectively use population information, making its overall convergence speed better than any QTS-based algorithm [39]. In contrast, the current-best update strategy may maintain excessive structural diversity in the population. In addition, due to the repair mechanism, the algorithm generates many solutions that have the same fitness value but different structures. This causes the algorithm to oscillate, making it even harder for the probabilities to converge, which greatly reduces both convergence speed and search ability. However, using the global best solution as a guide can effectively avoid this oscillation. It allows the qubit probabilities to converge steadily, keeps the algorithm working effectively, and significantly improves the overall performance.



(**a**) Scenario 1: $T_g = 0.3$. (**b**) Scenario 2: $T_g = 0.6$. (**c**) Scenario 3: $T_g = 0.9$.

Figure 4: Detection probability under uniform request distribution.

Table 2: Scenario 1 with connectivity constraint.

Deploy. Strat.	# Sens.	Satisf. (%)	Conn.
AEQTSwE	80 ± 1.41	100	OK
QTSwE-GB	80 ± 1.18	100	OK
QTSwE-CB [7]	83 ± 1.35	100	OK
QTSwE	83	100	OK
PFDA	87	100	OK
BDA	85 ± 1	99.72 ± 0.99	OK
Diff-Deploy	67	100	–
DDA	80	100	–
Min-Miss	87	98	–
Max-Min-Cov	63	100	–
Max-Avg-Cov	87	79.56	–
Grid	87	69.80	OK
Random-Uniform	87	88.70 ± 1.31	–
Random-Bernoulli	87	89.33 ± 1.96	–

Note: “Deploy. strat.” denotes deployment strategy; “# Sens.” denotes the number of sensors; “Satisf.” denotes the satisfaction rate; “Conn.” denotes connectivity.

Table 3: Scenario 1 without connectivity constraint.

Deploy. Strat.	# Sens.	Satisf. (%)
AEQTSwE	80 ± 1.33	100
QTSwE-GB	82 ± 1.62	100
QTSwE-CB [7]	93 ± 3.49	100
QTSwE	62	100
PFDA	63	100
BDA	63	97.35 ± 0.24
Diff-Deploy	63	99.60
DDA	63	94.68
Min-Miss	63	91.32

(Continued)

Table 3 (continued)

Deploy. Strat.	# Sens.	Satisf. (%)
Max-Min-Cov	63	100
Max-Avg-Cov	63	74.28
Grid	63	90.12
Random-Uniform	63	75.46 ± 1.72
Random-Bernoulli	63	76.84 ± 1.69

Note: "Deploy. strat." denotes deployment strategy; "# Sens." denotes the number of sensors; "Satisf." denotes the satisfaction rate.

Table 4: Scenario 2 with connectivity constraint.

Deploy. Strat.	# Sens.	Satisf. (%)	Conn.
AEQTSwE	107 ± 0.69	100	OK
QTSwE-GB	108 ± 0.93	100	OK
QTSwE-CB [7]	118 ± 2.26	100	OK
QTSwE	120	100	OK
PFDA	127	100	OK
BDA	127	98.87 ± 0.26	OK
Diff-Deploy	127	99.48	–
DDA	127	86.76	–
Min-Miss	127	92.88	OK
Max-Min-Cov	127	98.96	–
Max-Avg-Cov	127	71.36	OK
Grid	127	81.64	OK
Random-Uniform	127	76.07 ± 1.51	–
Random-Bernoulli	127	77.91 ± 1.45	–

Note: "Deploy. strat." denotes deployment strategy; "# Sens." denotes the number of sensors; "Satisf." denotes the satisfaction rate; "Conn." denotes connectivity.

Table 5: Scenario 2 without connectivity constraint.

Deploy. Strat.	# Sens.	Satisf. (%)
AEQTSwE	107 ± 1.07	100
QTSwE-GB	108 ± 1.71	100
QTSwE-CB [7]	117 ± 10.07	100
QTSwE	115	100
PFDA	126	100
BDA	126	97.55 ± 0.24
Diff-Deploy	126	99.28
DDA	126	85.96
Min-Miss	126	92.32

(Continued)

Table 5 (continued)

Deploy. Strat.	# Sens.	Satisf. (%)
Max-Min-Cov	126	98.68
Max-Avg-Cov	126	71.04
Grid	126	81.12
Random-Uniform	126	77.38 ± 1.35
Random-Bernoulli	126	77.24 ± 1.71

Note: "Deploy. strat." denotes deployment strategy; "# Sens." denotes the number of sensors; "Satisf." denotes the satisfaction rate.

Table 6: Scenario 3 with connectivity constraint.

Deploy. Strat.	# Sens.	Satisf. (%)	Conn.
AEQTSwE	216 ± 0.82	100	OK
QTSwE-GB	217 ± 1.14	100	OK
QTSwE-CB [7]	235 ± 2.20	100	OK
QTSwE	230	100	OK
PFDA	254	100	OK
BDA	254	90.31 ± 0.84	OK
Diff-Deploy	254	98.44	OK
DDA	254	92.32	–
Min-Miss	254	94.52	OK
Max-Min-Cov	254	98.76	OK
Max-Avg-Cov	254	69.04	OK
Grid	254	85.08	OK
Random-Uniform	254	72.84 ± 1.36	–
Random-Bernoulli	254	76.30 ± 1.42	–

Note: "Deploy. strat." denotes deployment strategy; "# Sens." denotes the number of sensors; "Satisf." denotes the satisfaction rate; "Conn." denotes connectivity.

Table 7: Scenario 3 without connectivity constraint.

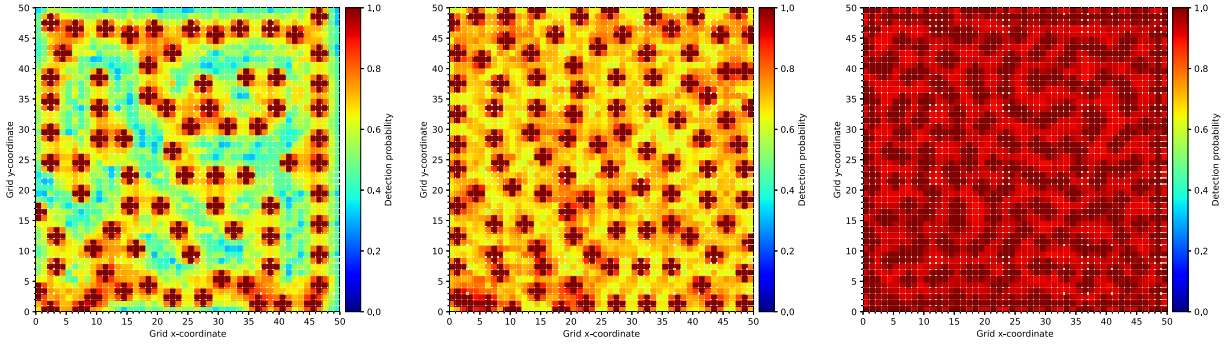
Deploy. Strat.	# Sens.	Satisf. (%)
AEQTSwE	216 ± 0.91	100
QTSwE-GB	217 ± 0.86	100
QTSwE-CB [7]	235 ± 2.22	100
QTSwE	220	100
PFDA	262	100
BDA	262	89.99 ± 0.81
Diff-Deploy	262	99.44
DDA	262	96.20
Min-Miss	262	95.24

(Continued)

Table 7 (continued)

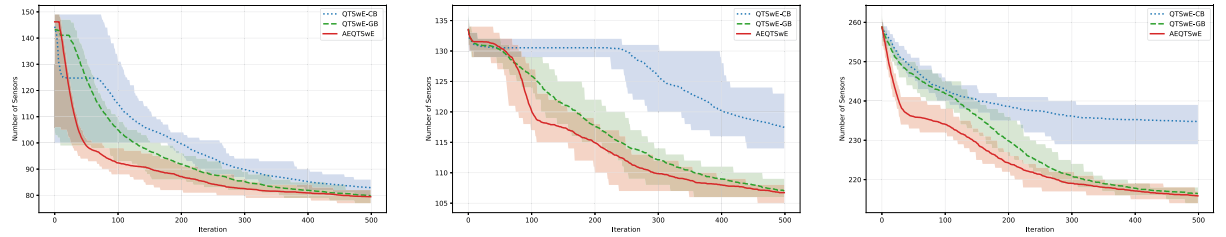
Deploy. Strat.	# Sens.	Satisf. (%)
Max-Min-Cov	262	99.80
Max-Avg-Cov	262	69.68
Grid	262	87.96
Random-Uniform	262	73.77 ± 1.33
Random-Bernoulli	262	78.69 ± 1.53

Note: "Deploy. strat." denotes deployment strategy; "# Sens." denotes the number of sensors; "Satisf." denotes the satisfaction rate.



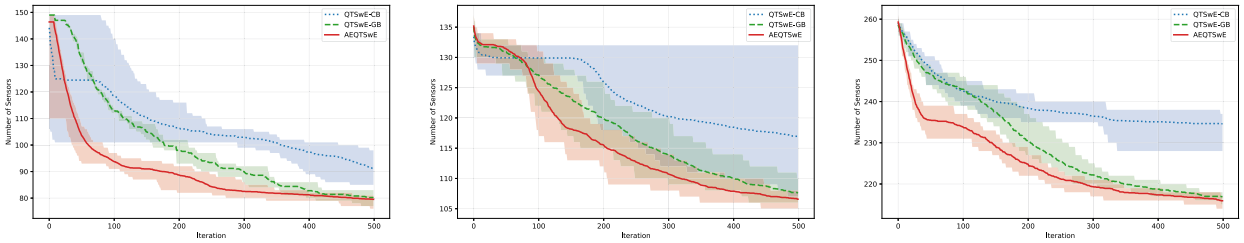
(**a**) Scenario 1: $T_g = 0.3$. (**b**) Scenario 2: $T_g = 0.6$. (**c**) Scenario 3: $T_g = 0.9$.

Figure 5: Sensor deployment of AEQTSwE across three uniform distribution scenarios.



(**a**) $T_g = 0.3$. (**b**) $T_g = 0.6$. (**c**) $T_g = 0.9$.

Figure 6: Convergence of QTSwE-CB [7], QTSwE-GB, and AEQTSwE across three **constrained** scenarios.



(**a**) $T_g = 0.3$. (**b**) $T_g = 0.6$. (**c**) $T_g = 0.9$.

Figure 7: Convergence of QTSwE-CB [7], QTSwE-GB, and AEQTSwE across three **unconstrained** scenarios.

Table 8: Detailed data comparison for Scenarios 1–3.

T_g	Conn.	Deploy. Strat.	Best	Worst	Mean	Std	Median
0.3	Y	AEQTSwE	77	82	80	1.41	79
0.3	Y	QTSwE-GB	77	82	80	1.18	80
0.3	Y	QTSwE-CB [7]	80	86	83	1.35	83
0.3	N	AEQTSwE	76	82	80	1.33	80
0.3	N	QTSwE-GB	77	84	82	1.62	82
0.3	N	QTSwE-CB [7]	86	99	93	3.49	91
0.6	Y	AEQTSwE	105	108	107	0.69	107
0.6	Y	QTSwE-GB	106	109	108	0.93	107
0.6	Y	QTSwE-CB [7]	114	123	118	2.26	117.5
0.6	N	AEQTSwE	105	108	107	1.07	106
0.6	N	QTSwE-GB	103	110	108	1.71	107
0.6	N	QTSwE-CB [7]	107	132	117	10.07	111
0.9	Y	AEQTSwE	214	217	216	0.82	216
0.9	Y	QTSwE-GB	214	218	217	1.14	216.5
0.9	Y	QTSwE-CB [7]	229	239	235	2.20	235
0.9	N	AEQTSwE	214	217	216	0.91	216
0.9	N	QTSwE-GB	215	218	217	0.86	217
0.9	N	QTSwE-CB [7]	228	238	235	2.22	236

Note: "Conn." indicates whether the connectivity constraint is considered, where 'Y' and 'N' denote yes and no, respectively; "Deploy. strat." denotes deployment strategy.

5.2 Non-Uniform Distribution

To simulate real sensing environments where different regions have different levels of importance, and following the experimental design of Kuo et al. [7], we construct the scenario shown in Fig. 8a. In this scenario, the sensing map is divided into several regions, each with its own detection threshold T_g: $T_g = 0.9$ in the central area, $T_g = 0.6$ in the surrounding area, and $T_g = 0.3$ in the outer region. We reproduce the distribution of T_g as closely as possible. From the deployment results in Fig. 8b,c, we observe that the outcomes with and without connectivity are very similar, with average sensor counts of 107 and 108. As summarized in Table 9, AEQTSwE also shows much better performance than the method of Kuo et al. [7] under the connectivity constraint. We further analyze the convergence behavior of the three methods, QTSwE-CB [7], QTSwE-GB, and AEQTSwE. As shown in Fig. 8d,e, AEQTSwE converges faster and achieves

the best final results among the compared variants in this scenario. Because AEQTSwE uses all population information, it can achieve consistent and stable performance on different problems. Its standard deviation is also lower than those of QTSwE-CB and QTSwE-GB.

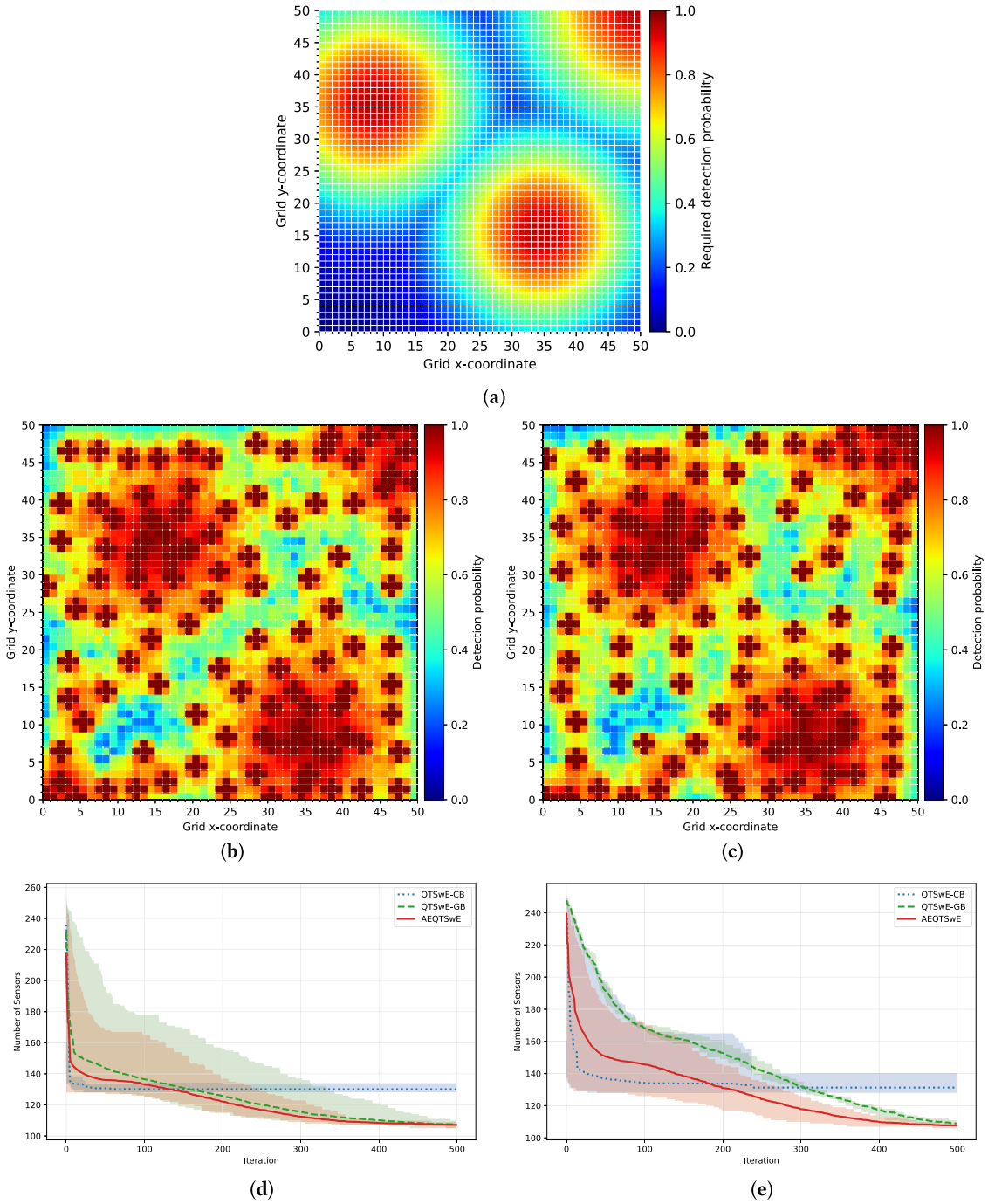


Figure 8: Heatmaps and convergence results for all Non-Uniform cases. (**a**) Detection probability under Non-Uniform distribution. (**b**) AEQTSwE deployment (Non-Uniform). (**c**) AEQTSwE deployment (Non-Uniform; no connectivity constraint). (**d**) Convergence comparison (Non-Uniform). (**e**) Convergence comparison (Non-Uniform, no connectivity constraint).

Table 9: Non-uniform distribution with connectivity constraint.

Deploy. Strat.	# Sens.	Satisf. (%)	Conn.
AEQTSwE	107 ± 0.98	100	OK
QTSwE-GB	108 ± 1.38	100	OK
QTSwE-CB [7]	131 ± 2.87	100	OK
QTSwE	137	100	OK
PFDA	149	100	OK
BDA	149	98.05 ± 0.28	OK
Diff-Deploy	149	95.96	OK
DDA	149	84.64	OK
Min-Miss	149	81.24	OK
Max-Min-Cov	149	76.04	OK
Max-Avg-Cov	149	76.96	OK
Grid	149	76.32	OK
Random-Uniform	149	72.25 ± 1.17	OK
Random-Bernoulli	149	74.56 ± 1.17	OK

Note: "Deploy. strat." denotes deployment strategy; "# Sens." denotes the number of sensors; "Satisf." denotes the satisfaction rate; "Conn." denotes connectivity.

Table 10 further shows the detailed statistical results of the three quantum-inspired methods under the Non-Uniform scenario with and without the connectivity constraint. Although AEQTSwE and QTSwE-GB obtain close results in this scenario, AEQTSwE still achieves slightly better basic statistical values. In particular, AEQTSwE uses fewer sensors than QTSwE-GB on average and also performs much better than QTSwE-CB. This shows that the proposed AEQTSwE remains effective under Non-Uniform sensing requirements, where different areas have different detection demands.

Table 10: Detailed data comparison for Non-Uniform distribution.

Conn.	Deploy. Strat.	Best	Worst	Mean	Std	Median
Y	AEQTSwE	105	109	107	0.98	107
Y	QTSwE-GB	105	110	108	1.38	107.5
Y	QTSwE-CB [7]	128	138	131	2.87	130
N	AEQTSwE	106	110	108	0.96	108
N	QTSwE-GB	108	114	109	1.33	109
N	QTSwE-CB [7]	119	156	127	8.28	123

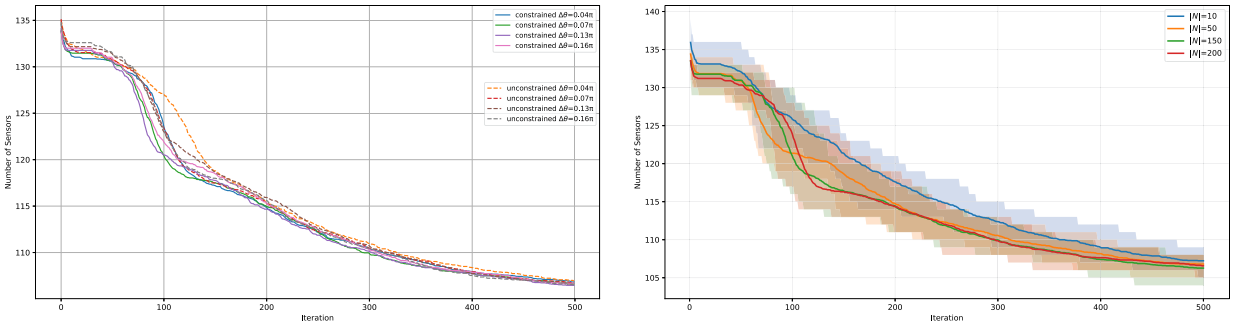
Note: "Conn." indicates whether the connectivity constraint is considered, where 'Y' and 'N' denote yes and no, respectively; "Deploy. strat." denotes deployment strategy.

5.3 Sensitivity Analysis and Ablation Study

In this subsection, we analyze the impact of different parameters in AEQTSwE, such as the rotation angle ($\Delta\theta$) and population size, to better understand its performance. We also discuss the effects of different algorithm components, including the repair mechanism, entanglement mechanism (for initial solutions), and the computational time spent on each component. This helps examine the optimization behavior of AEQTSwE in more detail. Finally, since some characteristics can be clearly observed in specific cases, certain experiments in this section are conducted only under fixed scenarios.

5.3.1 Parameter Sensitivity Analysis

Following the main experimental setting, we fix the maximum number of iterations to 500 and test the scenarios with and without the connectivity constraint at $T_g = 0.6$. First, we fix the population size to $|N| = 100$ and test different rotation angles ($\Delta\theta = 0.04\pi$, 0.07π, 0.13π, and 0.16π). As shown in Fig. 9a, the convergence speed and final results are very similar in both constrained and unconstrained scenarios. This is likely because the historical global-best solution set maintains good diversity. With $|N| = 100$, the population information can quickly correct the qubit states, leading to similar performance.



(**a**) Constrained/Unconstrained with different $\Delta\theta$.

(**b**) Constrained with different $|N|$.

Figure 9: Parameter sensitivity analysis for different $\Delta\theta$ and $|N|$ at $T_g = 0.6$.

In another experiment, we fix $\Delta\theta = 0.1\pi$ and test different population sizes: $|N| = 10$, 50, 150, and 200. Fig. 9b shows that the convergence is slower when $|N| = 10$. Although $|N| = 50$ is slightly faster, it is still slower than the larger sizes. For the other population sizes, the convergence speed and results are almost the same. This indicates that our setting of $|N| = 100$ is an appropriate size, as it can most efficiently utilize the information from the entire population.

Finally, the overall results in Fig. 9 show that AEQTSwE can maintain stable performance under different settings of $\Delta\theta$ and $|N|$. In particular, for the key parameter $\Delta\theta$, the results in Fig. 9a show that AEQTSwE does not have large fluctuations when $\Delta\theta$ changes. This indicates that AEQTSwE shows relatively low parameter sensitivity in this setting, which is consistent with the results reported in the original AEQTS paper [8]. However, the population-size results are slightly different from those reported in the original AEQTS paper [8]. In the experiments with different population sizes, as shown in Fig. 9b, the final convergence results are not very different between small and large populations. The main difference is only in the convergence speed, but this difference is also not large. This may be caused by the characteristics of this problem. Since the solutions may vary greatly, the fitness value is more likely to oscillate. As a result, the algorithm may not quickly converge toward one specific direction. Therefore, the effect of population size is not very significant in this problem.

5.3.2 Ablation Study

In this section, the algorithms are first tested in scenarios where $T_g = 0.3$, 0.6, and 0.9, with the connectivity constraint applied. To examine the effect of the repair mechanism, we first conduct an ablation experiment by disabling the repair procedure while keeping the other settings unchanged. The results under this no-repair setting are reported in Table 11. First, the repair mechanism is turned off to evaluate how removing it affects the optimization results and coverage rate. As shown in Table 11, compared to Table 8, some metrics show slight degradation and a small drop in stability. However, the difference in solution quality

for AEQTSwE is kept within 1 sensor, and the standard deviation does not change much. In addition, the coverage rate of AEQTSwE and the other two algorithms remains extremely close to 100%. This shows that although the repair mechanism guarantees a 100% coverage rate, further reduces the number of sensors, and improves algorithm stability, its benefits are relatively limited for a 50 × 50 grid problem size.

Table 11: Performance without the repair mechanism under different T_g with connectivity constraint.

T_g	Deploy. Strat.	Best	Worst	Mean	Std	Median	Avg. Coverage Rate
0.3	AEQTSwE	79	83	81	1.11	80	99.991%
	QTSwE-GB	79	87	82	1.74	82	99.964%
	QTSwE-CB [7]	91	111	100	5.00	99	100.000%
0.6	AEQTSwE	106	109	108	0.75	107	100.000%
	QTSwE-GB	107	111	109	1.09	108	100.000%
	QTSwE-CB [7]	135	151	147	4.23	147	100.000%
0.9	AEQTSwE	215	218	217	0.86	217	99.996%
	QTSwE-GB	214	219	217	1.22	217	99.999%
	QTSwE-CB [7]	260	275	268	3.85	267.5	100.000%

Note: "Deploy. strat." denotes deployment strategy.

To further analyze the actual contribution of the repair mechanism, an ablation study is designed with $T_g = 0.6$. In each iteration, 100 random solutions are generated and only repaired by the repair mechanism. The algorithm's quantum state update and evolution are completely skipped, and the fitness values are recorded for 500 iterations. This "purely random repair" baseline is plotted in Fig. 10 for comparison with AEQTSwE (both with and without the repair mechanism). Based on the convergence curves in Fig. 10, an interesting phenomenon can be observed: even when the repair mechanism is removed, AEQTSwE starts from a poor random space but still converges steadily to an excellent solution. This is due to its strong global exploration and local exploitation abilities. Although a 100% coverage rate cannot be mathematically guaranteed without the repair mechanism, the final coverage is still extremely close to 100%. On the other hand, the "purely random repair" baseline only slightly improves the solution at the beginning. It then stagnates because it lacks real evolutionary power, completely failing to reach the optimal solution. This phenomenon is consistent with the data in Tables 8 and 11. This result indicates that the repair mechanism mainly contributes by accelerating initial convergence and strictly ensuring constraint satisfaction.

In addition, the comparison between Tables 8 and 11 also shows that the repair mechanism has only a limited effect on the algorithms using the historical global-best update strategy, such as AEQTSwE and QTSwE-GB. However, it has a much larger effect on the algorithm using the current-best update strategy, such as QTSwE-CB [7]. This means that the repair mechanism may still have some influence, but its effect on AEQTSwE and QTSwE-GB is not very obvious in the 50 × 50 scenario. In fact, a more obvious difference can be observed in larger grid scenarios, as discussed in Section 5.4. Nevertheless, the effect of the update mechanism is still more important than that of the repair mechanism, and the key optimization ability to drive the process and escape local optima ultimately relies on the evolutionary mechanism of the AEQTSwE algorithm itself.

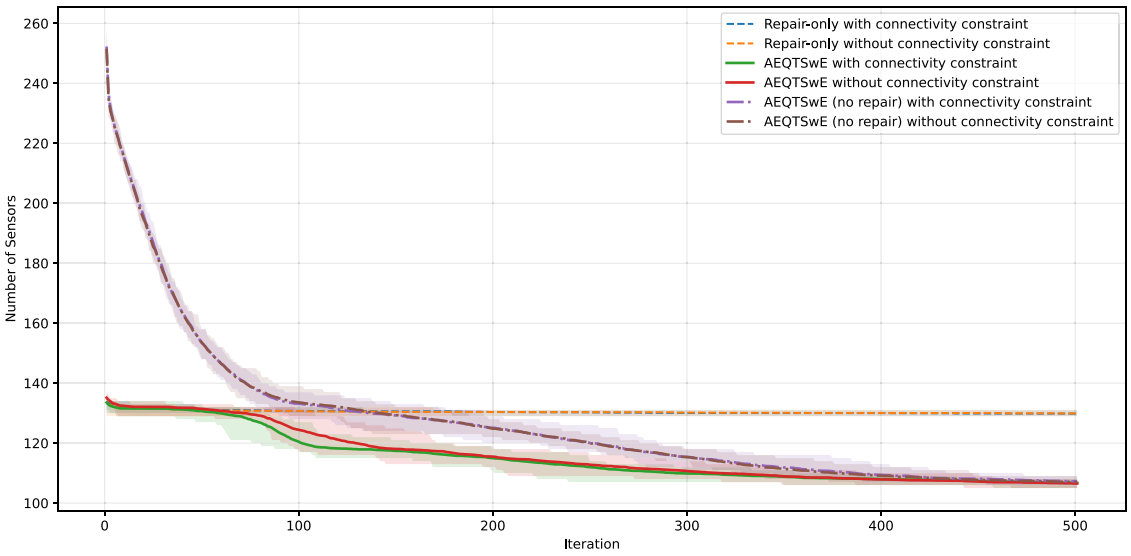
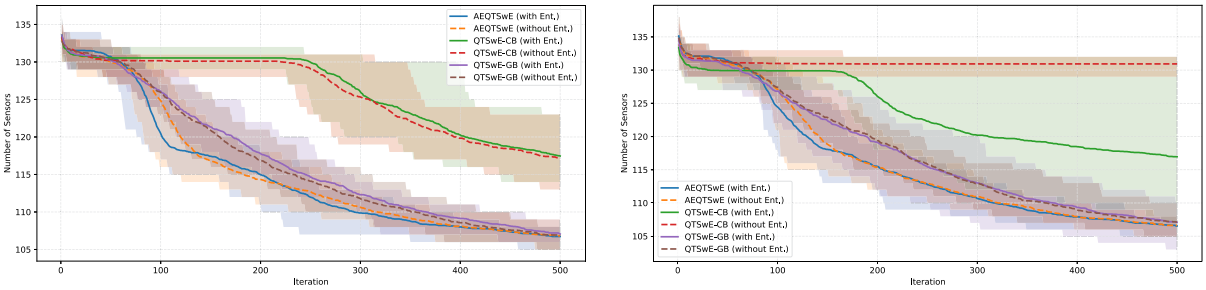


Figure 10: Ablation study on the repair mechanism.

Next, we also test the generation of initial solutions with and without the entanglement mechanism. The results are shown in Fig. 11. As shown in the figure, the effect of the entanglement mechanism is not very pronounced, but it still leads to slightly faster convergence. One possible reason is that the repair mechanism can also effectively reduce the number of sensors, so the differences between the initial solutions are not very large. Furthermore, Fig. 11b shows that QTSwE-CB [7] heavily relies on this entanglement mechanism. This is because the entanglement mechanism provides a more reasonable and fixed initial solution pattern, allowing QTSwE-CB [7] to start from relatively better initial solutions that are more suitable for its search process, thereby achieving better exploration and convergence. However, the tests on QTSwE-GB and AEQTSwE show that, by using the historical global-best solution set to update qubits, these methods no longer need to rely on the entanglement mechanism.



(**a**) With connectivity constraint. (**b**) Without connectivity constraint.

Figure 11: Ablation study on the entanglement mechanism at $T_g = 0.6$.

The execution time percentage of each component in AEQTSwE is also recorded. Since the current implementation is still a prototype written in Python, the execution speed is not yet optimized. After the feasibility and effectiveness of the algorithm are confirmed, lower-level programming languages or parallel computing can be further used to accelerate it. Therefore, this study analyzes the execution time distribution of different components as a basis for future optimization. Specifically, this study selects two scenarios

with the connectivity constraint for analysis, namely T_g = 0.9 and the Non-Uniform distribution. In the T_g = 0.9 scenario, the execution time of the entanglement mechanism is almost 0%, the repair mechanism accounts for about 38%, and the main search stage of AEQTSwE accounts for about 62%. In the Non-Uniform distribution scenario, the execution time of the entanglement mechanism is also almost 0%, the repair mechanism accounts for about 47%, and the main search stage of AEQTSwE accounts for about 53%. These results show that the entanglement mechanism is mainly used to generate initial solutions, and its time cost is almost negligible. Therefore, it does not introduce extra burden. In contrast, the repair mechanism accounts for about 40% to 50% of the total execution time. This is because this mechanism includes local search. Although it increases the computational cost, it also helps the algorithm find better solutions more quickly. On the other hand, the main search stage still accounts for the largest single portion of the total execution time because it requires 50,000 fitness evaluations (Table 1). However, these evaluations are mainly dominated by repeated coverage and connectivity checks rather than by the quantum-inspired update itself. Therefore, the most expensive low-level operations are the repair operations and the coverage/connectivity evaluation procedures. In the future, program optimization, better data structures, incremental coverage updates, faster connectivity checking, or parallel processing can effectively reduce the overall execution time and further improve the practicality of AEQTSwE for large-scale problems.

5.4 Scalability Analysis on Larger Grids

Two larger grid scenarios, 70 × 70 and 100 × 100, are also tested under the same setting of T_g = 0.6. Because the two scenarios have different grid sizes and evaluation purposes, the number of runs and the experimental settings are not exactly the same. The 70 × 70 scenario is run for 30 independent experiments with and without the connectivity constraint. This experiment is mainly used to evaluate the performance of the algorithms in a larger grid scenario. The 100 × 100 scenario is run for 10 independent experiments with the connectivity constraint. This scenario is used not only to compare the algorithms on a larger grid, but also to further evaluate the effect of the repair mechanism. Since the overall results show consistent trends among the algorithms, and considering the total runtime, this part is tested only under the "with connectivity" scenario.

Tables 12 and 13 show a consistent trend: methods using the historical global-best solution set outperform the current-best update strategy, and AEQTSwE achieves the most stable performance while maintaining full connectivity and a 100% average coverage rate. AEQTSwE still outperforms the other two algorithms when all sensors remain connected and the average coverage rate reaches 100%. Fig. 12 further shows the effect of the repair mechanism on these three algorithms in more detail. Compared with Fig. 10, it can be observed that, in this larger-scale scenario, the repair mechanism not only guarantees the coverage rate and connectivity but also removes redundant sensors. Therefore, it provides high-quality feasible solutions in the early stage of evolution, causing the number of sensors to decrease rapidly at the beginning. However, without the help of the repair mechanism, although AEQTSwE still outperforms the other two algorithms, it cannot converge to the same result as the version with the repair mechanism in the larger scenario. This also indicates that the repair mechanism can help stabilize the search process of AEQTSwE. In other words, the repair mechanism shows a scale-dependent delayed effect: its advantage becomes more evident as the grid size increases, helping the overall search process converge better. These results also indicate that AEQTSwE can maintain stable performance in these larger grid settings.

Table 12: Results for $T_g = 0.6$ on 70×70 and 100×100 grids.

Grid	Conn.	Deploy. Strat.	Best	Worst	Mean	Std	Median
70×70	Y	AEQTSwE	202	208	205	1.52	205
70×70	Y	QTSwE-GB	206	214	209	2.25	209
70×70	Y	QTSwE-CB [7]	252	257	254	1.28	254
70×70	N	AEQTSwE	202	207	205	1.48	204
70×70	N	QTSwE-GB	207	220	214	3.11	214
70×70	N	QTSwE-CB [7]	237	249	245	3.08	244
100×100	Y	AEQTSwE	411	431	419	5.37	419
100×100	Y	QTSwE-GB	432	447	440	4.43	440
100×100	Y	QTSwE-CB [7]	503	507	505	1.25	505

Note: "Conn." indicates whether the connectivity constraint is considered, where 'Y' and 'N' denote yes and no, respectively; "Deploy. strat." denotes deployment strategy.

Table 13: Results for $T_g = 0.6$ and a 100×100 grid with connectivity constraint (no repair).

Deploy. Strat.	Best	Worst	Mean	Std	Median	Avg. Coverage Rate
AEQTSwE	448	467	455	4.96	455	100%
QTSwE-GB	466	482	475	6.13	476	100%
QTSwE-CB [7]	631	647	643	4.69	643	99.72%

Note: "Deploy. strat." denotes deployment strategy.

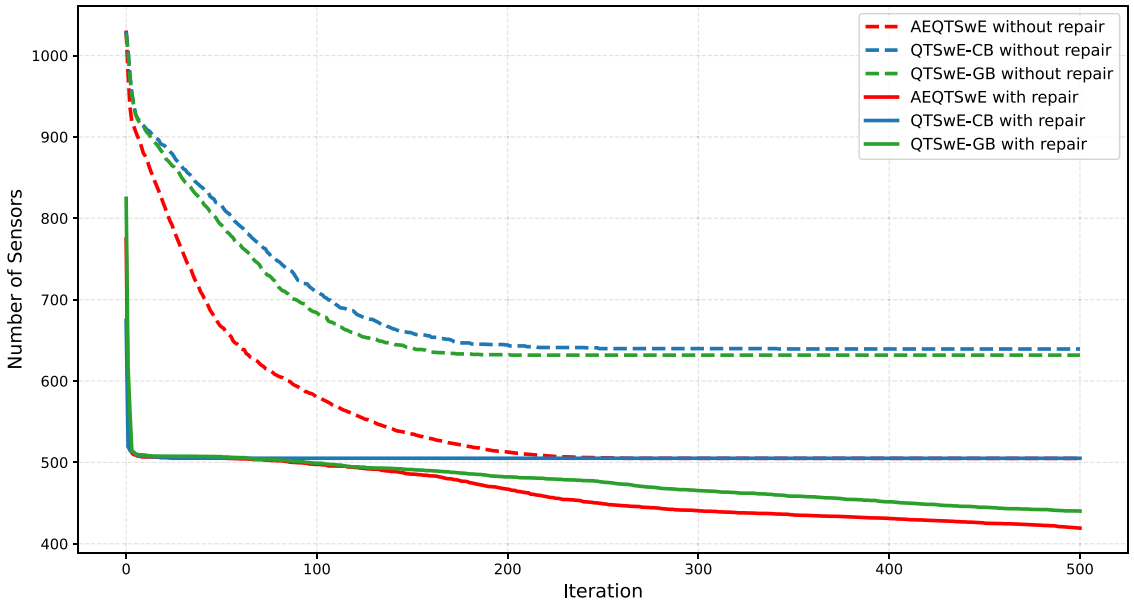


Figure 12: Scalability analysis on the 100×100 grid with and without the repair mechanism.

The above scalability results are also consistent with the runtime analysis and the complexity discussion in Section 4.3. Since the repair procedure and fitness evaluation account for the largest computational cost,

a larger grid is expected to increase the runtime. When the grid size increases from 50×50 to 100×100, this effect becomes more obvious because the total number of grid cells L grows quadratically with the side length of the grid. However, when the population size and the maximum number of iterations are fixed, the computational cost still grows polynomially. Therefore, the main scalability bottleneck comes from the repeated repair and evaluation process required to maintain feasible deployment solutions, rather than from the qubit update mechanism of AEQTSwE itself.

5.5 Statistical Significance Analysis

To further confirm whether the performance differences among different algorithms are statistically significant, this study discusses non-parametric statistical tests in addition to descriptive statistics, such as best, worst, mean, median, and standard deviation. Since the results of multiple independent runs of metaheuristic algorithms may not satisfy the normality assumption, this study uses the Kruskal–Wallis test [42], the Mann–Whitney U test [43], and the Friedman test [44] for statistical analysis. The significance level is set to 0.05. This allows a more accurate discussion of the differences among these independent runs and provides a better comparison of the overall performance of different algorithms.

5.5.1 Kruskal–Wallis Test [42]

First, this study applies the Kruskal–Wallis test [42] to examine whether there are significant differences among AEQTSwE, QTSwE-GB, and QTSwE-CB under the different benchmarks. The Kruskal–Wallis test [42] is suitable for comparing the distribution differences among three or more independent samples.

For the following three statistical tests, let the result sequence of the i-th method be $X_i = \{x_{i,1}, x_{i,2}, \ldots, x_{i,n_i}\}$, where n_i is the number of independent runs. In this study, the compared methods are tested using their independent-run results rather than only their average values.

All observations from the compared methods are first pooled and ranked together. Let RV_i denote the rank value, or rank sum, of the i-th method, and let $n_{\text{total}} = \sum_{i=1}^{k} n_i$ denote the total number of observations from all k methods. The Kruskal–Wallis statistic is then computed as

$$H = \frac{12}{n_{\text{total}}(n_{\text{total}}+1)} \sum_{i=1}^{k} \frac{RV_i^2}{n_i} - 3(n_{\text{total}}+1).$$

Under the null hypothesis that all methods come from the same distribution, H approximately follows a chi-square distribution with $k-1$ degrees of freedom. The corresponding p-value is then used to determine whether an overall significant difference exists.

The detailed test results are shown in Table 14. A p-value smaller than 0.05 indicates that the compared methods have a statistically significant overall difference. However, the Kruskal–Wallis test [42] does not identify which two methods are significantly different. Therefore, when the Kruskal–Wallis test [42] shows significance, the Mann–Whitney U test [43] is further used for pairwise comparison.

Table 14: Kruskal–Wallis test [42] results for individual scenarios.

T_g	Scen.	Grid	Conn.	RV_{A}	RV_{GB}	RV_{CB}	C	H_{c}	p-Value
0.3	U	50×50	Y	878	1026	2191	0.98	51.72	5.88×10^{-12}
0.3	U	50×50	N	627.5	1202.5	2265	0.99	68.34	1.44×10^{-15}
0.6	U	50×50	Y	846	984	2265	0.96	62.31	2.94×10^{-14}
0.6	U	50×50	N	829	1061.5	2204.5	0.98	54.01	1.87×10^{-12}

(Continued)

Table 14 (continued)

T_g	Scen.	Grid	Conn.	RV_A	RV_{GB}	RV_{CB}	C	H_c	p-Value
0.9	U	50 × 50	Y	781	1049	2265	0.97	62.76	2.35×10^{-14}
0.9	U	50 × 50	N	662.5	1167.5	2265	0.97	67.41	2.30×10^{-15}
0.6	U	70 × 70	Y	492.5	1337.5	2265	0.99	77.20	1.72×10^{-17}
0.6	U	100 × 100	Y	55	155	255	1.00	25.89	2.39×10^{-6}
–	N	50 × 50	Y	820	1010	2265	0.98	61.36	4.74×10^{-14}
–	N	50 × 50	N	629.5	1200.5	2265	0.98	68.88	1.11×10^{-15}

Note: RV_A, RV_{GB}, and RV_{CB} denote the rank values of AEQTSwE, QTSwE-GB, and QTSwE-CB, respectively. Due to the higher computational cost, AEQTSwE was run for 10 independent experiments in the 100 × 100 scenario. "Scen." indicates the scenario type, where 'U' denotes Uniform and 'N' denotes Non-uniform. "Conn." indicates whether the connectivity constraint is considered; 'Y' and 'N' denote yes and no, respectively.

5.5.2 Mann–Whitney U Test [43]

The Mann–Whitney U test [43] was used to further examine whether two methods have a significant difference. For any problem setting, assume that the two methods to be compared are method A and method B. Their results from multiple independent runs can be expressed as

$$X_A = \{x_{A,1}, x_{A,2}, \ldots, x_{A,n_A}\}, \quad X_B = \{x_{B,1}, x_{B,2}, \ldots, x_{B,n_B}\},$$

where n_A and n_B denote the number of experimental runs of method A and method B, respectively. The Mann–Whitney U test [43] first combines the two groups of results into one sample set with $n_A + n_B$ samples and ranks all samples. If several results have the same value, the average rank is also used to handle tied ranks.

After ranking, let the rank values RV_A and RV_B denote the rank sums of method A and method B. Then, the Mann–Whitney U statistics of the two groups can be calculated from the rank sums as

$$U_A = RV_A - \frac{n_A(n_A+1)}{2}, \quad U_B = RV_B - \frac{n_B(n_B+1)}{2}.$$

Since $U_A + U_B = n_A n_B$, this study uses $U = \min(U_A, U_B)$ as the test statistic and calculates the corresponding p-value based on U. If tied ranks exist, the corresponding tie correction is used when calculating the p-value.

This study performs two pairwise comparisons for each problem setting. Only the comparisons between AEQTSwE and the other algorithms are considered, namely "AEQTSwE vs. QTSwE-GB" and "AEQTSwE vs. QTSwE-CB". The detailed results are shown in Table 15. Most comparisons show statistically significant differences. Only a few cases, including the comparisons with QTSwE-GB when $T_g = 0.3$, $T_g = 0.6$, and the Non-Uniform scenarios, do not show significant differences. This is reasonable because AEQTSwE and QTSwE-GB have very close performance in these cases. However, by further checking the descriptive results in Tables 8 and 10, AEQTSwE still shows slightly better basic statistical values in these non-significant cases. This implies that, under a detailed comparison, the AEQTS-based [8] update remains generally more effective than the QTS-based [39] update in the tested WSN deployment scenarios.

Table 15: Pairwise Mann–Whitney U test [43] results.

T_g	Scen.	Grid	Conn.	Pair	U	p-Value
0.3	U	50 × 50	Y	AEQTSwE vs. QTSwE-GB	376	2.65×10^{-1}
0.3	U	50 × 50	Y	AEQTSwE vs. QTSwE-CB	37	7.30×10^{-10}
0.3	U	50 × 50	N	AEQTSwE vs. QTSwE-GB	162.5	1.43×10^{-5}
0.3	U	50 × 50	N	AEQTSwE vs. QTSwE-CB	0	2.20×10^{-11}
0.6	U	50 × 50	Y	AEQTSwE vs. QTSwE-GB	381	2.77×10^{-1}
0.6	U	50 × 50	Y	AEQTSwE vs. QTSwE-CB	0	1.13×10^{-11}
0.6	U	50 × 50	N	AEQTSwE vs. QTSwE-GB	353.5	1.44×10^{-1}
0.6	U	50 × 50	N	AEQTSwE vs. QTSwE-CB	10.5	6.07×10^{-11}
0.9	U	50 × 50	Y	AEQTSwE vs. QTSwE-GB	316	3.98×10^{-2}
0.9	U	50 × 50	Y	AEQTSwE vs. QTSwE-CB	0	1.64×10^{-11}
0.9	U	50 × 50	N	AEQTSwE vs. QTSwE-GB	197.5	9.69×10^{-5}
0.9	U	50 × 50	N	AEQTSwE vs. QTSwE-CB	0	1.94×10^{-11}
0.6	U	70 × 70	Y	AEQTSwE vs. QTSwE-GB	27.5	3.44×10^{-10}
0.6	U	70 × 70	Y	AEQTSwE vs. QTSwE-CB	0	2.27×10^{-11}
0.6	U	100 × 100	Y	AEQTSwE vs. QTSwE-GB	0	1.78×10^{-4}
0.6	U	100 × 100	Y	AEQTSwE vs. QTSwE-CB	0	1.70×10^{-4}
–	N	50 × 50	Y	AEQTSwE vs. QTSwE-GB	355	1.50×10^{-1}
–	N	50 × 50	Y	AEQTSwE vs. QTSwE-CB	0	2.10×10^{-11}
–	N	50 × 50	N	AEQTSwE vs. QTSwE-GB	164.5	1.18×10^{-5}
–	N	50 × 50	N	AEQTSwE vs. QTSwE-CB	0	1.93×10^{-11}

Note: Due to the higher computational cost, AEQTSwE was run for 10 independent experiments in the 100 × 100 scenario. "Scen." indicates the scenario type, where 'U' denotes Uniform and 'N' denotes Non-uniform. "Conn." indicates whether the connectivity constraint is considered; 'Y' and 'N' denote yes and no, respectively.

5.5.3 Friedman Test [44]

Finally, this study further uses the Friedman test [44] to analyze the overall ranking differences among the three methods across different T_g problems. The Friedman test [44] is a rank-based non-parametric test. It is suitable for the overall comparison of multiple methods on the same set of benchmarks. Unlike the Kruskal–Wallis test [42], the Friedman test [44] treats each benchmark as a block. It ranks all methods within each block, and then checks whether there is an overall significant difference among the methods based on the ranks across all blocks.

Let B denote the number of benchmarks, and let k denote the number of compared methods. Let $r_{b,j}$ denote the rank of the j-th method on the b-th benchmark, where $b = 1, 2, \ldots, B$ and $j = 1, 2, \ldots, k$. The rank sum of the j-th method over all benchmarks can be expressed as $RV_j = \sum_{b=1}^{B} r_{b,j}$, and its average rank is $\bar{r}_j = \frac{RV_j}{B}$.

The statistic of the Friedman test [44] can then be expressed as

$$\chi_F^2 = \frac{12}{Bk(k+1)} \sum_{j=1}^{k} RV_j^2 - 3B(k+1).$$

Next, the corresponding p-value is calculated based on χ_F^2 and the degrees of freedom $k-1$. Under the chi-square approximation, χ_F^2 approximately follows a chi-square distribution with $k-1$ degrees of freedom. Therefore, $p = 1 - F_{\chi^2_{k-1}}(\chi_F^2)$, where $F_{\chi^2_{k-1}}(\cdot)$ denotes the cumulative distribution function of the chi-square distribution with $k-1$ degrees of freedom.

Table 16 shows that AEQTSwE obtains the best rank in the selected benchmark settings, QTSwE-GB ranks second, and QTSwE-CB ranks third. Therefore, AEQTSwE has the lowest average rank across all settings. The p-value is 4.5399929×10^{-5}, indicating that the overall ranking differences among the three methods across the problem settings are statistically significant. This further supports the observations from the Kruskal–Wallis test [42], the Mann–Whitney U test [43], and the experimental tables, showing that AEQTSwE has better overall performance in most tested WSN deployment scenarios.

Table 16: Friedman test [44] results across the selected benchmark settings.

T_g	Scen.	Grid	Conn.	AEQTSwE	QTSwE-GB	QTSwE-CB
0.3	U	50 × 50	Y	1	2	3
0.3	U	50 × 50	N	1	2	3
0.6	U	50 × 50	Y	1	2	3
0.6	U	50 × 50	N	1	2	3
0.9	U	50 × 50	Y	1	2	3
0.9	U	50 × 50	N	1	2	3
0.6	U	70 × 70	Y	1	2	3
0.6	U	100 × 100	Y	1	2	3
–	N	50 × 50	Y	1	2	3
–	N	50 × 50	N	1	2	3
			Average rank	1.0	2.0	3.0
			Overall p-value		4.5399929×10^{-5}	

Note: Due to the higher computational cost, AEQTSwE was run for 10 independent experiments in the 100 × 100 scenario. “Scen.” indicates the scenario type, where ‘U’ denotes Uniform and ‘N’ denotes Non-uniform. “Conn.” indicates whether the connectivity constraint is considered; ‘Y’ and ‘N’ denote yes and no, respectively.

6 Conclusion

This study addresses the WSN deployment problem by combining the AEQTS algorithm with WSN-specific fitness functions and repair mechanisms. Following the initialization strategy of Kuo et al. [7], we propose the AEQTSwE framework, which aims to satisfy three objectives simultaneously: the event detection threshold, topology connectivity, and the minimization of sensor count. Under the conceptual design of AEQTS [8], the update process uses information from all solutions in the population, allowing the algorithm to perform global search and local refinement more effectively.

Experimental results show that, across eight Uniform and Non-Uniform distributions in the 50 × 50 grid setting, AEQTSwE achieves better performance than many existing heuristic and metaheuristic methods [7,37] in terms of sensor usage and convergence speed. Furthermore, the average sensor count is reduced by about 10.6%, the median by about 9.6%, and the standard deviation by about 61.5%. This indicates that AEQTSwE not only has strong optimization ability, but is also more stable than the method proposed by Kuo et al. [7]. It also maintains coverage and connectivity effectively, demonstrating its ability to handle the WSN deployment problem more efficiently.

Additional experiments on the algorithm itself show that AEQTSwE is not highly sensitive to the rotation angle or the population size. Moreover, the ablation experiments show that the main optimization ability comes from the evolutionary update mechanism, while the repair mechanism mainly supports constraint satisfaction, early-stage search, and larger-scale convergence. The repair mechanism also shows a delayed effect, as its advantage becomes more evident at larger scales and helps AEQTSwE achieve better convergence. The statistical tests also show that the performance differences among the compared stochastic variants are generally significant, and AEQTSwE achieves better results in most tested benchmark settings. The complexity and runtime analyses show that the main scalability bottleneck comes from repeated repair and evaluation, rather than from the quantum-inspired update itself. The overall complexity grows polynomially, so the performance can be further improved by optimizing the repair or evaluation procedures.

Overall, this study shows that incorporating a historical global-best ensemble into the quantum-inspired update process provides a simple yet effective way to enhance both convergence stability and solution quality. The proposed framework maintains a relatively low-parameter structure while achieving competitive and stable optimization performance, making it a practical approach for the tested WSN deployment problems.

7 Future Work

Several directions can be explored to further extend this work. First, the current implementation is based on a Python prototype. Performance can be further improved through optimized implementations using lower-level programming languages or parallel computing techniques. Second, the proposed AEQTSwE framework can be extended to more complex and realistic deployment settings, including heterogeneous sensors, obstacle-aware environments, dynamic deployment scenarios, and real-world geographical maps containing physical obstacles and dynamic constraints, as explored in recent metaheuristic studies [9,10]. Third, more rigorous convergence analysis and more efficient repair/evaluation implementations should be investigated. Since the complexity and runtime analyses indicate that repair and evaluation dominate the computational cost, future work should focus on incremental coverage updates, faster connectivity checking, and parallel computation.

Acknowledgement: The authors used ChatGPT and Claude to assist with language polishing, grammar correction, and improving the clarity and consistency of mathematical and statistical descriptions during manuscript preparation. All scientific content was independently verified, derived, and revised by the authors.

Funding Statement: This research was funded by the National Science and Technology Council (NSTC), Taiwan, under Grant NSTC 114-2221-E-197-010-.

Author Contributions: The authors confirm their contributions to the paper as follows: Conceptualization, Kuo-Chun Tseng; methodology, Kuo-Chun Tseng; software, I-Chia Chen; validation, I-Chia Chen and Yu-Chieh Cho; formal analysis, Kuo-Chun Tseng and I-Chia Chen; investigation, I-Chia Chen and Yu-Chieh Cho; resources, Kuo-Chun Tseng; data curation, I-Chia Chen and Yu-Chieh Cho; writing—original draft preparation, Kuo-Chun Tseng and Yu-Chieh Cho; writing—review and editing, Kuo-Chun Tseng; visualization, I-Chia Chen and Yu-Chieh Cho; supervision, Kuo-Chun Tseng; project administration, Kuo-Chun Tseng; funding acquisition, Kuo-Chun Tseng. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The authors confirm that the data supporting the findings of this study are available within the article. Additional data related to the experimental results are available from the corresponding author upon reasonable request.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript

ABC	Artificial Bee Colony
ACO	Ant Colony Optimization
AEQTSwE	Amplitude-Ensemble QTS with Entanglement initialization using global-best (historical global-best) update strategy
AI	Artificial Intelligence
BDA	Bernoulli Deployment Algorithm
DDA	Differentiated Deployment Algorithm
DE	Differential Evolution
Diff-Deploy	Differentiated Deployment
E2BDA	Efficient Evidence-Based sensor Deployment Algorithm
EBDA	Evidence-Based Deployment Algorithm
GA	Genetic Algorithm
GWO	Grey Wolf Optimizer
HGPSO	Hybrid Genetic Particle Swarm Optimization
HSA	Harmony Search Algorithm
IoT	Internet of Things
Max-Avg-Cov	Maximize Average Coverage
Max-Min-Cov	Maximize Minimum Coverage
Max-Miss	Maximize Miss probability
ML	Machine Learning
Min-Miss	Minimize Miss probability
PFDA	Potential Field Deployment Algorithm
PSO	Particle Swarm Optimization
QTS	Quantum-inspired Tabu Search algorithm
QTSwE-CB	QTS with Entanglement initialization using Current-Best (best-per-iteration) Update Strategy
QTSwE-GB	QTS with Entanglement initialization using Global-Best (historical global-best) Update Strategy
RL	Reinforcement Learning
SPP	Sensor Placement Problem
WSN/WSNs	Wireless Sensor Network/Wireless Sensor Networks

References

1. Lai CS, Jia Y, Dong Z, Wang D, Tao Y, Lai QH, et al. A review of technical standards for smart cities. Clean Technol. 2020;2(3):290–310.
2. Abdulwahid HM, Mishra A. Deployment optimization algorithms in wireless sensor networks for smart cities: a systematic mapping study. Sensors. 2022;22(14):5094. doi:10.3390/s22145094.
3. Du R, Santi P, Xiao M, Vasilakos AV, Fischione C. The sensable city: a survey on the deployment and management for smart city monitoring. IEEE Commun Surv Tutor. 2018;21(2):1533–60.
4. Zhang J, Yan T, Son SH. Deployment strategies for differentiated detection in wireless sensor networks. In: Proceedings of the 2006 3rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks; 2006 Sep 25–28; Reston, VA, USA. New York, NY, USA: IEEE; 2006. p. 316–25.
5. Dey P, Balachandran N, Chatterjee D. Complexity of constrained sensor placement problems for optimal observability. Automatica. 2021;131(4):109758. doi:10.1016/j.automatica.2021.109758.
6. Srivastava A, Mishra PK. A survey on WSN issues with its heuristics and meta-heuristics solutions. Wirel Pers Commun. 2021;121(1):745–814. doi:10.1007/s11277-021-08659-x.
7. Kuo SY, Chou YH, Chen CY. Quantum-inspired algorithm for cyber-physical visual surveillance deployment systems. Comput Netw. 2017;117(1):5–18. doi:10.1016/j.comnet.2016.11.013.

8. Tseng KC, Lai WC, Chen IC, Hsiao YH, Chiue JY, Huang WC. Amplitude-ensemble quantum-inspired tabu search algorithm for solving 0/1 knapsack problems. In: Proceedings of the 2024 IEEE International Conference on Systems, Man, and Cybernetics (SMC); 2024 Oct 6–10; Kuching, Malaysia. New York, NY, USA: IEEE; 2024. p. 718–25.
9. Anka F. A novel hybrid metaheuristic method for efficient decentralized IoT network layouts. Internet Things. 2025;32(10):101612. doi:10.1016/j.iot.2025.101612.
10. Priyadarshi R. Efficient node deployment for enhancing coverage and connectivity in wireless sensor networks. Sci Rep. 2025;15(1):29052.
11. Zou Y, Chakrabarty K. Uncertainty-aware sensor deployment algorithms for surveillance applications. In: Proceedings of the GLOBECOM'03. IEEE Global Telecommunications Conference (IEEE Cat. No. 03CH37489); 2003 Dec 1–5; San Francisco, CA, USA. New York, NY, USA: IEEE; 2003. p. 2972–6.
12. Zou Y, Chakrabarty K. Uncertainty-aware and coverage-oriented deployment for sensor networks. J Parallel Distrib Comput. 2004;64(7):788–98.
13. Senouci MR, Mellouk A, Oukhellou L, Aissani A. Uncertainty-aware sensor network deployment. In: Proceedings of the 2011 IEEE Global Telecommunications Conference-GLOBECOM 2011; 2011 Dec 5–9; Houston, TX, USA. New York, NY, USA: IEEE; 2011. p. 1–5.
14. Yick J, Mukherjee B, Ghosal D. Wireless sensor network survey. Comput Netw. 2008;52(12):2292–330. doi:10.1016/j.comnet.2008.04.002.
15. More A, Raisinghani V. A survey on energy efficient coverage protocols in wireless sensor networks. J King Saud Univ Comput Inf Sci. 2017;29(4):428–48.
16. Farsi M, Elhosseini MA, Badawy M, Ali HA, Eldin HZ. Deployment techniques in wireless sensor networks, coverage and connectivity: a survey. IEEE Access. 2019;7:28940–54.
17. Deif DS, Gadallah Y. Classification of wireless sensor networks deployment techniques. IEEE Commun Surv Tutor. 2013;16(2):834–55.
18. Priyadarshi R, Gupta B, Anurag A. Deployment techniques in wireless sensor networks: a survey, classification, challenges, and future research issues. J Supercomput. 2020;76(9):7333–73. doi:10.1007/s11227-020-03166-5.
19. Egwuche OS, Singh A, Ezugwu AE, Greeff J, Olusanya MO, Abualigah L. Machine learning for coverage optimization in wireless sensor networks: a comprehensive review. Ann Oper Res. 2023;91–92(8):1–67. doi:10.1007/s10479-023-05657-z.
20. Dhillon SS, Chakrabarty K. Sensor placement for effective coverage and surveillance in distributed sensor networks. In: Proceedings of the 2003 IEEE Wireless Communications and Networking, WCNC 2003; 2003 Mar 16–20; New Orleans, LA, USA. New York, NY, USA: IEEE; 2003. p. 1609–14.
21. Reda SM, Abdelhamid M, Latifa O, Amar A. Efficient uncertainty-aware deployment algorithms for wireless sensor networks. In: Proceedings of the 2012 IEEE Wireless Communications and Networking Conference (WCNC); 2012 Apr 1; Paris, France. New York, NY, USA: IEEE; 2012. p. 2163–7.
22. Senouci MR, Mellouk A, Oukhellou L, Aissani A. WSNs deployment framework based on the theory of belief functions. Comput Netw. 2015;88(4):12–26. doi:10.1016/j.comnet.2015.05.014.
23. Li Y, Gao W, Wu C, Wang Y. Deployment of sensors in WSN: an efficient approach based on dynamic programming. Chin J Electron. 2015;24(1):33–6. doi:10.1049/cje.2015.01.006.
24. Ni Q, Du H, Pan Q, Cao C, Zhai Y. An improved dynamic deployment method for wireless sensor network based on multi-swarm particle swarm optimization. Nat Comput. 2017;16(1):5–13.
25. ZainEldin H, Badawy M, Elhosseini M, Arafat H, Abraham A. An improved dynamic deployment technique based on genetic algorithm (IDDT-GA) for maximizing coverage in wireless sensor networks. J Ambient Intell Humaniz Comput. 2020;11(10):4177–94. doi:10.1007/s12652-020-01698-5.
26. Ayinde BO, Barnawi AY. Differential evolution based deployment of wireless sensor networks. In: Proceedings of the 2014 IEEE/ACS 11th International Conference on Computer Systems and Applications (AICCSA); 2014 Nov 10–13; Doha, Qatar. New York, NY, USA: IEEE; 2014. p. 131–7.
27. Wang Z, Xie H. Wireless sensor network deployment of 3D surface based on enhanced grey wolf optimizer. IEEE Access. 2020;8:57229–51. doi:10.1109/access.2020.2982441.

28. Gorkemli B, Al-Dulaimi Z. On the performance of quick artificial bee colony algorithm for dynamic deployment of wireless sensor networks. Turk J Electr Eng Comput Sci. 2019;27(6):4038–54. doi:10.3906/elk-1101-1030.
29. Deif DS, Gadallah Y. An ant colony optimization approach for the deployment of reliable wireless sensor networks. IEEE Access. 2017;5:10744–56. doi:10.1109/access.2017.2711484.
30. Al-Fuhaidi B, Mohsen AM, Ghazi A, Yousef WM. An efficient deployment model for maximizing coverage of heterogeneous wireless sensor network based on harmony search algorithm. J Sens. 2020;2020:8818826.
31. Saad A, Senouci MR, Benyattou O. Toward a realistic approach for the deployment of 3D wireless sensor networks. IEEE Trans Mob Comput. 2020;21(4):1508–19. doi:10.1109/tmc.2020.3024939.
32. Wang Z, Xie H, Hu Z, Li D, Wang J, Liang W. Node coverage optimization algorithm for wireless sensor networks based on improved grey wolf optimizer. J Algorithms Comput Technol. 2019;13:1–15.
33. Dokeroglu T, Canturk D, Kucukyilmaz T. A survey on pioneering metaheuristic algorithms between 2019 and 2024. arXiv:2501.14769. 2024.
34. Houssein EH, Saad MR, Djenouri Y, Hu G, Ali AA, Shaban H. Metaheuristic algorithms and their applications in wireless sensor networks: review, open issues, and challenges. Clust Comput. 2024;27(10):13643–73. doi:10.1007/s10586-024-04619-9.
35. Hussein NK, Qaraad M, El Najjar AM, Farag M, Elhosseini MA, Mirjalili S, et al. Schrödinger optimizer: a quantum duality-driven metaheuristic for stochastic optimization and engineering challenges. Knowl-Based Syst. 2025;328:114273.
36. Ibrahim MQ, Qaraad M, Hussein NK, Farag M, Guinovart D. Secant optimization algorithm for efficient global optimization. Sci Rep. 2026;16:6659.
37. Aitsaadi N, Achir N, Boussetta K, Pujolle G. Artificial potential field approach in WSN deployment: cost, QoM, connectivity, and lifetime constraints. Comput Netw. 2011;55(1):84–105. doi:10.1016/j.comnet.2010.07.017.
38. Grover LK. A fast quantum mechanical algorithm for database search. In: Proceedings of the 28th Annual ACM Symposium on Theory of Computing; 1996 May 22–24; Philadelphia, PA, USA. p. 212–9.
39. Chou YH, Chiu CH, Yang YJ. Quantum-inspired tabu search algorithm for solving 0/1 knapsack problems. In: Proceedings of the Genetic and Evolutionary Computation Conference; 2011 Jul 11–16; Dublin, Ireland. p. 55–6.
40. Chiang HP, Chou YH, Chiu CH, Kuo SY, Huang YM. A quantum-inspired Tabu search algorithm for solving combinatorial optimization problems. Soft Comput. 2014;18(9):1771–81.
41. Moreno-Saavedra LM, Costa VG, Garrido-Saez A, Jiménez-Fernández S, Portilla-Figueras JA, Salcedo-Sanz S. Evolutionary optimization of spatially-distributed multi-sensors placement for indoor surveillance environments with security levels. Future Gener Comput Syst. 2025;166(1–2):107727. doi:10.1016/j.future.2025.107727.
42. Kruskal WH, Wallis WA. Use of ranks in one-criterion variance analysis. J Am Stat Assoc. 1952;47(260):583–621. doi:10.1080/01621459.1952.10483441.
43. Mann HB, Whitney DR. On a test of whether one of two random variables is stochastically larger than the other. Ann Math Stat. 1947;18(1):50–60. doi:10.1214/aoms/1177730491.
44. Friedman M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. J Am Stat Assoc. 1937;32(200):675–701. doi:10.1080/01621459.1937.10503522.