# A compact Black Hole Algorithm for localization of mobile sensor network

**Wei-Min Zheng**
Shandong University of Science and Technology
**Shi-Lei Xu**
Shandong University of Science and Technology
**Jeng-Shyang Pan**
Shandong University of Science and Technology
**Qing-Wei Chai** ( mimanxiaowei@163.com )
Shandong University of Science and Technology https://orcid.org/0000-0001-5587-0589
**Pei Hu**
Shandong University of Science and Technology

**Research Article**

**Keywords:** Mobile node localization, Black Hole algorithm, Compact strategy, Monte Carlo Localization

**Posted Date:** April 12th, 2022

**DOI:** https://doi.org/10.21203/rs.3.rs-1343477/v1

**Noname manuscript No.**
(will be inserted by the editor)

# A compact Black Hole Algorithm for localization of mobile sensor network

**Wei-Min Zheng · Shi-Lei Xu · Jeng-Shyang Pan · Qing-Wei Chai* · Pei Hu**

Received: date / Accepted: date

**Abstract** In mobile wireless sensor networks, localization accuracy and cost are the key issues to be considered. This paper is committed to solving the localization problem of mobile sensor networks. We propose an improved Black Hole (BH) algorithm based on compact strategy and elitist learning strategy. An improved Monte Carlo localization (IMCL) algorithm based on multi-hop combines the novel algorithm to deal with the localization problem of mobile sensor networks. The performance of the novel algorithm is verified on 28 test functions of CEC 2013 and compared with other standard optimization algorithms. The results reveal that the novel algorithm has first-class performance. In the simulation experiment, the novel algorithm and several optimization algorithms are applied to IMCL. The comparison results show that the new heuristic algorithm combined with IMCL can provide more competitive results in mobile node localization.

**Keywords** Mobile node localization · Black Hole algorithm · Compact strategy · Monte Carlo Localization

## 1 Introduction

There are hundreds or thousands of sensor nodes in the objective area of Wireless Sensor Network (WSN), which is composed of processor, sensor, communication, power supply, and other modules to monitor the phenomenon in the deployment area [32, 29]. The information gathered is transmitted to the base

F. Author
College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China
E-mail: mimanxiaowei@163.com

S. Author
second address

station. However, due to lacking the location of the node, the transmitted information is meaningless. Therefore, localization of sensor nodes is the critical and fundamental attribute in WSN.

The sensor nodes of WSN consist of anchor (seed) nodes and unknown nodes. The former is usually equipped with a global positioning system (GPS) [40] module, while the latter is only used to collect information and lacks positioning functions. Because of economic and energy costs, as well as there are many unknown nodes in WSN, it is a challenging problem to estimate the location of unknown nodes. In recent years, various localization algorithms are proposed for the localization of unknown nodes, which are mainly divided into two categories: range-based localization algorithm [5, 20] and range-free localization algorithm [31, 41]. In the range-based localization algorithm, due to the low cost of measuring the Received Signal Strength (RSS), Patwari et al. [25] used RSS for localization without considering the error caused by the fading channel. Hofmann et al. [9] introduced the method of localization using Time Of Arrival (TOA). In [28], Savvides et al. mentioned a method for localization using the Time Difference Of Arrival (TDOA) of two different signals. Niculescu et al. [24] proposed a method about Angle Of Arrival (AOA) in 2003, assuming that each node has the ability of AOA, and all nodes can determine their direction and location in the ad-hoc network where only a small number of nodes have the ability of localization. The range-free localization algorithm mainly depends on the connectivity of nodes to locate unknown nodes. It first calculates the connection of nodes in the network, and then starts to estimate the locations of nodes.

Affected by the influence of external forces (wind and ocean current), nodes are always inevitably moving. Therefore, in many cases, we must regard the topology of WSN as dynamic. The continuous relocation of unknown nodes leads to the increase of network energy consumption, and the network delay also greatly affects the positioning accuracy. At the same time, Doppler shift [12] also affects the localization of mobile nodes, and occurs when the signal transmitter moves relative to the receiver. At present, many localization algorithms have been proposed for mobile node localization, such as particle filter-based algorithm [36] and Monte Carlo localization (MCL) algorithm [8, 34].

Meta-heuristic algorithms [35, 11] mainly refer to algorithms that simulate various natural phenomena, such as Ant Colony Optimization (ACO) algorithm [6] inspired by ant foraging phenomenon, black hole (BH) algorithm [15, 4] inspired by natural black hole phenomenon, Phasmatodea (stick insect) population evolution (PPE) algorithm [33] inspired by the evolution of stick insect populations in nature, Cat Swarm Optimization (ASO) algorithm [7] inspired by the tracking and seeking behavior of cats in nature, and Cuckoo Search (CS) algorithm [37] inspired by cuckoo parasitism and hatching behavior. Since the meta-heuristic algorithm can effectively solve various optimization problems, people pay more and more attention to it in recent years. From the early Particle Swarm Optimization (PSO) algorithm [10] and Ge-

netic Algorithm (GA) [26] to BH algorithm and CS algorithm, more and more optimization algorithms have been proposed.

In recent years, people try to use meta-heuristic algorithms to optimize various problems of WSN, and have made great progress. For example, in [14], Gupta et al. tried to use the CS algorithm to optimize the clustering protocol in WSN and save energy. In [17], Kulkarni et al. introduced several applications of PSO algorithm in WSN, such as node localization and data aggregation. In [30], Sharmin et al. used the ACO algorithm to optimize the data transmission path in WSN to reduce energy waste. In order to prolong the lifetime of WSN, Arjunan et al. [1] proposed a hybrid routing protocol based on the ACO algorithm. Gumaida et al. [13] combined PSO with Variable Neighborhood Search (VNS) algorithm to improve the positioning accuracy of WSN. Chai et al. [2] proposed parallel Whale Optimization (PWOA) algorithm to optimize the localization of WSN. In [42], Zheng et al. combined the compact strategy into the adaptive PSO algorithm and proposed the Compact Adaptive Particle Swarm Optimization (CAPSO) algorithm, which was applied to the localization of mobile nodes. In [3], Chai et al. presented an improved Fish Migration Optimization (FMO) algorithm, which was applied in solving the localization problem of Wireless Sensor Network (WSN) on 3-D terrain and achieved good results.

The BH algorithm mentioned above was proposed for data clustering. In this paper, the Compact Black Hole (CBH) algorithm is propesed, combined with the elitist learning strategy (ELS) to improve the ability of BH to jump out of the local optimum. At the same time, compact strategy is combined with BH algorithm to make it suitable for memory-constrained scenarios, such as WSN. This paper also improves MCL algorithm, called IMCL algorithm. Compared with the traditional MCL algorithm, IMCL algorithm has better localization accuracy and stability in mobile sensor networks.

The rest of the paper is organized as follows. Section 2 introduces the basic principles of BH algorithm and MCL algorithm. In Section 3, we introduce the strategies added to the BH algorithm and the improvements of MCL algorithm. Section 4 shows the performance of CBH algorithm, and compares CBH algorithm with several common optimization algorithms. In Section 5, we introduce the comparison between CBH and several common optimization algorithms in IMCL application. Finally, we summarize this paper in Section 6.

## 2 Related work

This section first introduces the basic principles of the BH algorithm, and then introduces the positioning process of the MCL algorithm.

### 2.1 Black Hole algorithm

Abdolreza Hatamlou proposed BH algorithm in 2013, which is inspired by the black hole phenomenon in the universe. In this algorithm, the global best individual is named black hole, which attracts other individuals to move towards it. Nevertheless, once the distance between an individual and the black hole is smaller than the event horizon, this individual would be swallowed and randomly generates a new individual in search space. Like GA or PSO, the BH algorithm evolves the population to an optimal solution through specific strategies. If the fitness of an individual is better than the optimal solution after moving, it will replace the optimal solution. However, the location update formula in BH is simpler and only related to the location of the optimal solution. The pseudocode is shown as Algorithm 1:

**Algorithm 1** BH

1: **Initialization:** Generate the population of stars randomly in search space
2: **while** (t < MaxGeneration) **do**
3: Calculate the fitness of each star
4: Select the star with the best fitness value as the black hole
5: Change the position of the star according to Eq. (1)
6: If the fitness value of a star is better than that of black hole, it will exchange position with the black hole
7: If the star is within the event horizon of the black hole, a new star is randomly generated in the search space to replace the original star
8: t = t + 1

In the initialization stage, the BH algorithm randomly generates a population in the search space, and selects the individual with the best fitness as the black hole and the rest as stars. In the iterative process, all-stars move towards the black hole, and the position movement formula of stars is as follows:

$$x_i(t+1) = x_i(t) + rand \times (x_{BH} - x_i(t)) \quad i = 1, 2, ..., N \tag{1}$$

where $x_i(t)$ and $x_i(t+1)$ are the locations of $i$th star at iterations $t$ and $t+1$, respectively. $x_{BH}$ is the location of black bole, $rand$ is a random number in the interval [0,1], and $N$ is the number of stars (candidate solutions). The event horizon radius influence the balance of exploration and exploitation of the BH algorithm, and the following equation defines it:

$$R = \frac{f_{BH}}{\sum_{i=1}^{N} f_i} \tag{2}$$

where $f_{BH}$ is the fitness of black hole and $f_i$ is the fitness of the $i$th star. $N$ is the size of population and is also the number of candidate of solutions.

## 2.2 Monte Carlo Localization

In mobile sensor network localization, nodes may leave the previous location and reach the current location. In order to facilitate prediction, we assume that the time is discrete. We need to relocate the node in each time unit to obtain the current position of the node. The observations from anchor nodes can help us filter some wrong predicted positions. The main idea of the MCL algorithm is shown as follows:

**Algorithm 2** MCL

1: **Initialization:** Initializing anchor nodes and unknown nodes, $N$ is a constant indicating the number of samples to be maintained, $L_0 = \{$set of random locations in the deployment area$\}$
2: **Step:** Solve the current set of possible positions $L_t$ according to location of node in last time unit $L_{t-1}$ and the new observation $o_t$, $L_t = \{\}$
3: **while** (size$(L_t) < N$) **do**

$$R = \left\{ l_t^i \mid l_t^i \text{ is selected from } p\left(l_t \mid l_{t-1}^i\right), l_{t-1}^i \in L_{t-1}, 1 \leqslant i \leqslant N \right\} \tag{3}$$

$$R_{\text{fitered}} = \left\{ l_t^i \mid l_t^i \text{ where } l_t^i \in R \text{ and } \ p\left(o_t \mid l_t^i\right) > 0 \right\} \tag{4}$$

$$L_t = \text{choose}\left(L_t \cup R_{\text{fittered}}, N\right) \tag{5}$$

Here, $l_t$ represents the possible position of the node at time t, $o_t$ represents the observation of the anchor node from time $t-1$ to time $t$, $p\left(l_t \mid l_{t-l}\right)$ represents the possible position prediction of the node at time $t$ according to the position at time $t-1$, and $p\left(l_t \mid o_t\right)$ indicates the possibility that the current node is at $l_t$ according to the observed value $o_t$. Therefore, we can use $l_{t-1}$ to predict the position distribution of nodes at time $t$, and then filter the impossible positions according to $o_t$. The set $L_t$ of n samples represents the distribution $l_t$, and the algorithm recursively calculates the sample set in each time unit, therefore, $L_{t-1}$ reflects all previous observations, which allows us apply $L_{t-1}$ and $o_t$ to calculate $l_t$.

### *2.2.1 Prediction*

In the prediction stage, the node applies the movement model to each sample through a set of possible positions $L_{t-1}$ obtained in the previous step, so as to obtain a new set of possible positions $L_t$. We do not know the speed and direction of the node movement, only know that the moving speed of the node is less than $Vmax$. If $l_{t-1}^i$ is the estimated position of the node in the $t-1$ iteration, the current position of node could be located in a circle with $l_{t-1}^i$ as the center and $Vmax$ as the radius.

*2.2.2 Filtering*

At this stage, the node filters the location obtained in the prediction stage through the new observation results. Here, we assume that the message can be received immediately. Figure 1 shows that there are four possible scenarios for anchor nodes, as shown below:

(1) outsiders: the anchor node that neither the last time unit nor this time unit has been monitored.
(2) arrivers: the anchor node whose last time unit is not monitored but this time unit is monitored.
(3) leavers: the anchor node whose last time unit is monitored but this time unit is not monitored.
(4) insiders: the anchor nodes that both last time unit and this time unit are monitored.

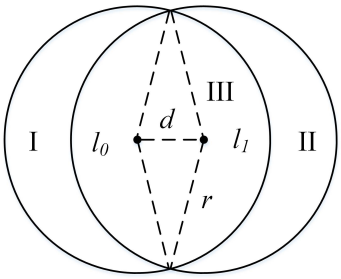



**Fig. 1** The anchor node is at $l_0$ at time 0 and moves to $l_1$ at time 1. The anchor node is *insider* for the nodes in region III, for the nodes in region II is *arriver*, for the nodes in region I is *leaver*, for all other region is *outsider*.

However, if we only rely on the direct information from the anchor node, the node will not know the previous location of the *arriver* or the current location of the *leaver*. Therefore, *arriver* and *leaver* are most useful for node positioning because they provide information when nodes enter and leave. We can let the neighbor node spread the information about the anchor node location (the collection of all anchor nodes and their locations monitored in the last time unit), which enables the node to find the information of the external anchor nodes. The node knows that it is not within the distance $r$ of any *outsider* anchor node, but it must be within the $2r$ of any anchor node monitored by its neighbors.

Figure 2 shows the filter conditions of *insider* and *outsider*. Let $S$ represent the set of all anchor nodes monitored by $N$, and $T$ represent the set of all anchor nodes monitored by $N$'s neighbors but not monitored and heard by $N$. Then the filter condition of the current position $l$ of the node is:

$$filter(l) = \forall s \in S, d(l,s) \leq r \wedge \forall s \in T, r < d(l,s) \leq 2r \tag{6}$$

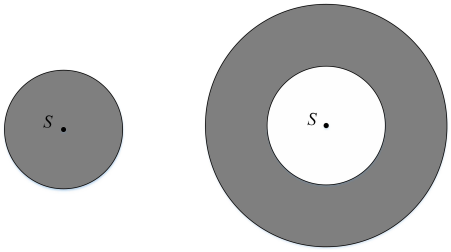



**Fig. 2** If the anchor node can directly monitor the node, the distance between them is within $r$. If the node is monitored by the neighbor of the anchor node but not by the anchor node, the distance between them is within ($r$,2$r$].

## 3 CBH and IMCL algorithm

In this section, we mainly introduce the improvements to BH and MCL algorithms. Firstly, the compact BH algorithm is proposed, which has more extraordinary searchability than the original algorithm and supports memory-constrained scenarios. Then the ELS is applied to the BH algorithm to improve the global searchability of the novel algorithm. Finally, the multi-hop is introduced in the MCL algorithm, which improves the localization accuracy of unknown nodes in WSN with a low anchor node density.

### 3.1 CBH algorithm

Unlike the traditional evolutionary algorithm, the essence of the estimated distribution algorithm (EDA) is a new stochastic optimization algorithm. It combines mathematical statistics with an evolutionary algorithm, and uses a statistical learning method to establish a probability model to describe population distribution. Population evolution of EDA is realized by updating the probability model.

We use a perturbation vector (PV) to represent the probability model of the population. In CBH algorithm, PV is an n × 2 matrix.

$$PV^t = [\mu^t, \sigma^t] \tag{7}$$

Where $\mu$ is the mean value of the perturbation vector, $\sigma$ is the standard deviation of the perturbation vector, and $t$ represents the current number of iterations. Each pair of $\mu$ and $\sigma$ corresponds to a Gaussian Probability Distribution Function (PDF). The PDF is truncated in [-1,1] and the amplitude is normalized to ensure the area is 1, as shown in Figure 3.

Through Chebyshev polynomials, PDF is used to construct a Cumulative Distribution Function (CDF) with values ranging from 0 to 1. The calculation formula of CDF is as follows:

$$CDF = \int_{-1}^{x} \mathrm{PDF} dx = \int_{-1}^{x} \frac{\sqrt{\frac{2}{\pi}} e^{-\frac{(x-\mu)^2}{2\delta^2}}}{\delta \left( \mathrm{erf}\left(\frac{\mu+1}{\sqrt{2}\delta}\right) - \mathrm{erf}\left(\frac{\mu-1}{\sqrt{2}\delta}\right)\right)} dx \tag{8}$$

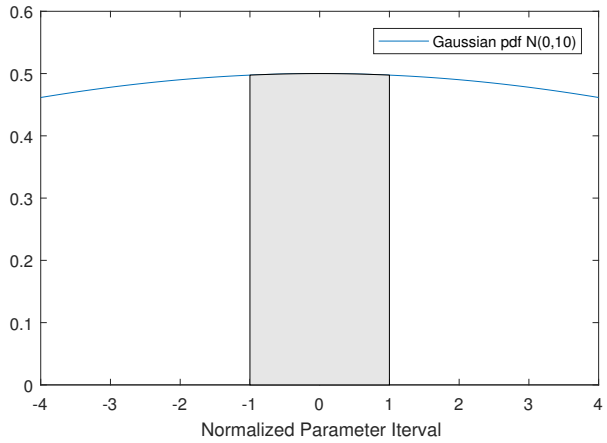



**Fig. 3** Normalization of truncated Gaussian curves (with standard deviation $\sigma = 10$)

Where $erf$ is the error function, $\mu$ and $\sigma$ are the mean and standard deviation in PV, respectively. The candidate solution is generated by using the inverse CDF, and two solutions are obtained after one iteration. The solution with a better fitness value is represented as a *winner*, and the other is a *loser*. Then, $\mu$ and $\sigma$ is updated respectively through the following formulas:

$$\mu_i^{t+1} = \mu_i^t + \frac{1}{N_p}\left( winner_i - loser_i \right) \tag{9}$$

$$\sigma_i^{t+1} = \sqrt{\left(\sigma_i^t\right)^2 + \left(\mu_i^t\right)^2 - \left(\mu_i^{t+1}\right)^2 + \frac{1}{N_p}\left(winner_i^2 - loser_i^2\right)} \tag{10}$$

In Eq. (9), $N_p$ represents the size of virtual population. The pseudocode of compact is shown as Algorithm 3.

**Algorithm 3** compact

```
while (t < maxGeneration) do
    Generate a solution x1 using the PV and inverse CDF
    Update x1 using the position update formula to get x2
    winner and loser are selected by comparing x1 and x2
    Update PV with winner and loser
    t = t + 1
```

Based on the compact algorithm introduced above, the compact BH algorithm can be designed. In the initialization phase, we set $\mu$ to 0 and $\sigma$ to a larger positive constant $\lambda$ ($\lambda = 10$). Setting $\sigma$ to 10 allows the algorithm to sample randomly throughout the interval. In the BH algorithm, the position of the black hole ($BH$) represents the current optimal solution. The pseudocode of CBH is shown as Algorithm 4.

**Algorithm 4** CBH

**Initialization:** $\mu = 0$, $\sigma = 10$, initialize $BH$
**while** (t < maxGeneration) **do**
    The candidate solution $x_1$ is generated by $\mu$ and $\sigma$
    **if** fitness($x_1$) > fitness($BH$) **then**
        $BH = x_1$
    Update $x_1$ to get $x_2$ by Eq. (1)
    Compete($x_2$,$BH$), choose *winner* and *loser* from $x_2$ and $BH$
    Update $\mu$ and $\sigma$ according to the Eq. (9) and Eq. (10)
    **if** fitness($x_2$) > fitness($BH$) **then**
        $BH = x_2$
    t = t + 1

It is worth noting that the BH algorithm is easy to fall into local optimum, so the actual performance of the algorithm is not very good. Firstly, CBH does not use one particle like the traditional compact algorithm but uses four particles to divide the original problem interval into four parts of the same length. After each particle runs on its interval for a while, the four particles iterate on the whole interval together. In this way, compared with a particle compact algorithm, it can effectively jump out of the local optimum. Secondly, as shown in Eq. (11), the parameter $w$ is added to the position update formula of CBH. The initial value of $w$ is 2 and gradually decreases to 1 with the progress of the iteration. CBH has a strong exploration ability at the beginning of the iteration, and then the exploitation ability of CBH is gradually enhanced with the operation of the algorithm. Compared with the BH algorithm, CBH can find the optimal solution faster or better.

$$x_{t+1} = x_t + w \times rand \times (x_{BH} - x_t) \tag{11}$$

where $t$ is the current iteration, $x_{BH}$ is the current global optimal solution.

Finally, the CBH algorithm adopts ELS in the convergence state, and ELS acts on the global optimal particle (BH). The local optimal solution may have some of the same structure as the global optimal solution which should be protected. Therefore, ELS acts randomly on a dimension of BH. ELS is carried out through Gaussian disturbance, and the specific formula is shown as follows:

$$BH^d = BH^d + (x^d_{max} - x^d_{min}) \times Gaussian(\mu, \sigma^2) \tag{12}$$

where $d$ is the randomly selected dimension, $x_{min}$ and $x_{max}$ are the lower and upper bounds of the problem, and $Gaussian(\mu, \sigma^2)$ is a random number. In this paper, $\mu$ is set to $BH^d$ and $\sigma^2$ is set to 3. The results show that these three methods effectively improve the ability of the algorithm to jump out of the local optimum, and properly balance the exploration and exploitation ability of the algorithm.

### 3.2 Improved Monte Carlo algorithm

The traditional MCL algorithm uses available anchor nodes one hop and two hops away from the unknown nodes for localization. The anchor node position

is initialized randomly, so it may be far away from the unknown node, which means that the anchor node has no meaning and is called an unusable anchor node. In the IMCL algorithm, more hops are used for positioning unknown nodes, which can further improve the positioning accuracy. Especially when the number of anchor nodes is small or the deployment area is large, multi-hop can improve the positioning accuracy to a greater extent. At the same time, it also brings the problem of longer positioning time. Therefore, it is challenging to balance the number of hops and positioning time. The IMCL algorithm in this paper uses the sum of the distances between the unknown nodes and the available anchor nodes as the evaluation index of the WSN hop count. The details are as follows:

$$Max\_Hop(i) = Sum\_Dist(i)/Vmax \tag{13}$$

where $Max\_Hop(i)$ represents the maximum number of hops suitable for unknown node $i$, $Sum\_Dist(i)$ refers to the sum of the distance between the $i$th unknown node and the usable anchor node, and $Vmax$ is the maximum speed of node movement.

The algorithm is mainly divided into two parts. In the forecasting phase, to estimate the location of the unknown node, the place information of neighbor nodes (contain anchor nodes and unknown nodes within one hop range) is necessary. When multiple nodes simultaneously detect an unknown node, the unknown node is located in the intersection of the communication areas of these nodes. There are N points sampled randomly in the common area, and the average position of these N points is the estimated position of the unknown node. This process is shown in Figure 4.

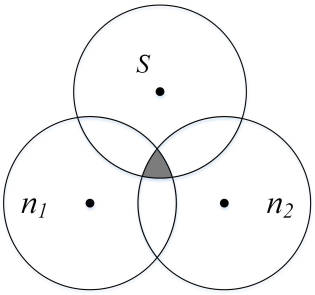



**Fig. 4** $S$ is the anchor node, $n_1$ and $n_2$ are unknown nodes that monitor the unknown nodes.

When the approximate position of the unknown node is known, the sum of the distances between the unknown node and the available anchor nodes can be obtained through calculation, and $Max\_Hop(i)$ can also be solved. Then we use Eq. (4) to locate unknown nodes, as shown in Figure 5.

Through comparative experiments in section 4.3, we can conclude that the IMCL algorithm has higher positioning accuracy than the original MCL algorithm.

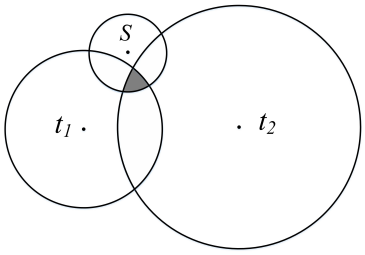



**Fig. 5** $S$ is the anchor node, $t_1$ is the anchor node two hops away from the unknown node, and $t_2$ is the anchor node three hops away from the unknown node.

## 4 Experimental analysis of algorithm

### 4.1 Benchmark functions and algorithm parameters

This section uses 28 benchmark functions in CEC 2013 [18] to test the actual performance of the CBH algorithm. CEC 2013 contains 5 unimodal functions, 15 basic multimodal functions, and 8 composition functions, which can comprehensively test the performance of the algorithm. In this experiment, the dimension of the test function is set to 20. We first test the performance of the original BH algorithm and CBH algorithm on 28 benchmark functions of CEC 2013 in Section 4.2, and then test the comparison of CBH algorithm with classical PSO and GA algorithm, as well as other common algorithms such as bat algorithm (BA) [38], sine cosine algorithm (SCA) [21] and whale optimization algorithm (WOA) [22] in Section 4.3. Finally, the performance of MCL algorithm and IMCL algorithm are compared, and the experimental results are analyzed in Section 4.4.

### 4.2 Comparion with the original BH algorithm

The performance of the BH and CBH algorithm is measured at a significant level at $\alpha = 0.05$ under Wilcoxon's signed rank test. Symbol ($<$) indicates that the performance of the BH algorithm is inferior to that of the CBH algorithm, symbol ($>$) indicates that the performance of the CBH algorithm is poor, and symbol ($=$) indicates that the performance of the two algorithms is similar. The specific results are shown in Table 1.

**Table 1** Comparision of CBH with the BH. Each algorithm is measured under Wilcoxon's signed rank test with the significant level $\alpha = 0.05$.

| Function | BH | | CBH |
|---|---|---|---|
| $f_1$ | 5.461E+03 | (<) | -1.400E+03 |
| $f_2$ | 2.902E+07 | (<) | 4.374E+06 |
| $f_3$ | 1.375E+14 | (<) | 9.460E+08 |
| $f_4$ | 4.267E+04 | (<) | 3.771E+04 |
| $f_5$ | 4.503E+02 | (<) | -9.999E+02 |
| $f_6$ | 3.913E+02 | (<) | -8.672E+02 |
| $f_7$ | 1.676E+04 | (<) | 4.626E+02 |
| $f_8$ | -6.791E+02 | (<) | -6.792E+02 |
| $f_9$ | -5.779E+02 | (=) | -5.774E+02 |
| $f_{10}$ | 1.845E+02 | (<) | -4.983E+02 |
| $f_{11}$ | -1.061E+02 | (<) | -3.800E+02 |
| $f_{12}$ | 5.504E+00 | (<) | -7.476E+01 |
| $f_{13}$ | 1.214E+02 | (<) | 5.059E+01 |
| $f_{14}$ | 4.011E+03 | (<) | 2.610E+02 |
| $f_{15}$ | 3.749E+03 | (<) | 2.603E+03 |
| $f_{16}$ | 2.016E+02 | (<) | 2.014E+02 |
| $f_{17}$ | 5.897E+02 | (<) | 3.398E+02 |
| $f_{18}$ | 6.916E+02 | (>) | 7.231E+02 |
| $f_{19}$ | 1.874E+03 | (<) | 5.044E+02 |
| $f_{20}$ | 6.096E+02 | (=) | 6.097E+02 |
| $f_{21}$ | 1.669E+03 | (<) | 1.050E+03 |
| $f_{22}$ | 5.602E+03 | (<) | 1.373E+03 |
| $f_{23}$ | 5.694E+03 | (<) | 4.859E+03 |
| $f_{24}$ | 1.292E+03 | (<) | 1.266E+03 |
| $f_{25}$ | 1.410E+03 | (<) | 1.364E+03 |
| $f_{26}$ | 1.405E+03 | (<) | 1.401E+03 |
| $f_{27}$ | 2.336E+03 | (<) | 2.258E+03 |
| $f_{28}$ | 4.802E+03 | (=) | 4.984E+03 |

According to the data in Table 1, we can conclude that CBH algorithm performs well on 24 functions, and its performance on the $f_9$, $f_{20}$, $f_{28}$ functions is similar to the BH algorithm, except that on the $f_{18}$, its performance is not as good as that of the BH algorithm. Therefore, CBH algorithm has outstanding performance in unimodal function and good performance in multimodal function, which shows that the strategy adopted for the CBH algorithm is successful, improves the ability of the algorithm to jump out of local optimization and convergence performance, and enable CBH to find better results faster under the same number of iterations.

### 4.3 Comparion with the common optimization algorithm

This section compares the CBH algorithm with other common optimization algorithms. The parameter settings of the CBH algorithm are the same as those in the previous section, and the parameter settings of other optimization

algorithms are shown in Table 2. The comparison results with CBH algorithm are shown in Table 3.

**Table 2** Parameter settings for each related algorithm.

| Name | Parameter |
|---|---|
| PSO | $N_p = 30$, $\phi_1 = -0.2$, $\phi_2 = -0.07$, $\phi_3 = 3.74$, $\lambda_1 = 1$, $\lambda_2 = 1$ |
| GA | $N_p = 30$, mutation rate = 0.01, crossover rate = 0.9 |
| BA | $N_p = 30$, loudness = 0.6, pulse rate = 0.7, $f_{min} = 0$, $f_{max} = 1$ |
| WOA | $N_p = 30$, $a = 2$, $a_2 = -1$ |
| SCA | $N_p = 30$, $a = 2$, $r_2 \in [0, 2\pi]$, $r_3 \in [0, 2]$, $r_4 \in [0, 1]$ |

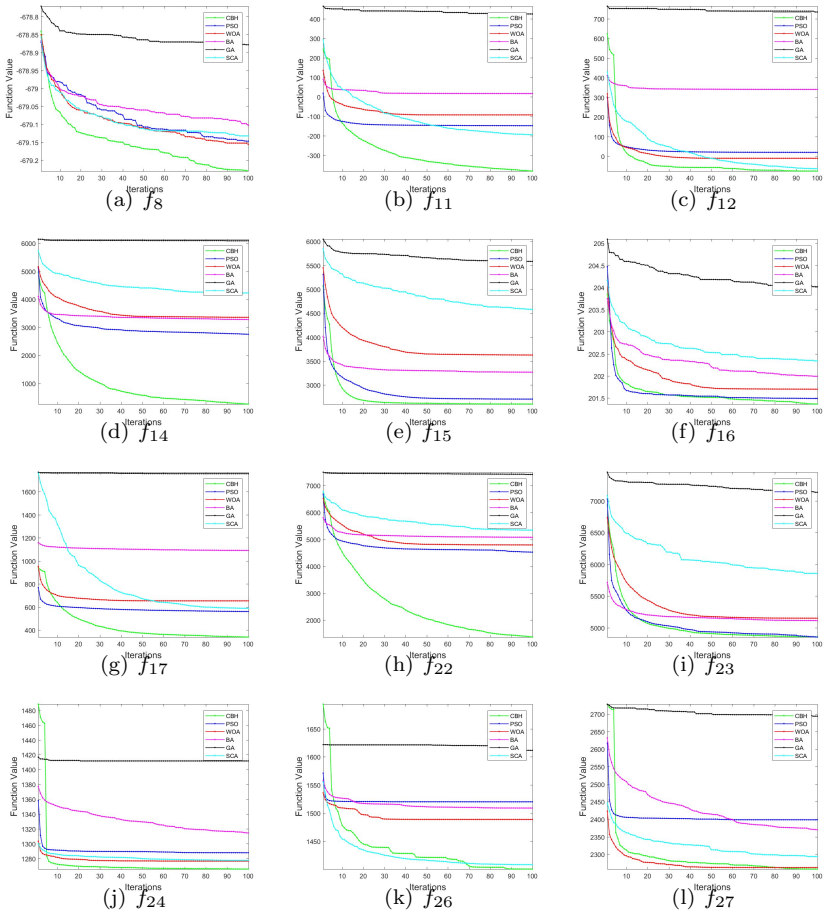
**Table 3** Comparision of CBH with the BA, PSO, SCA, WOA. Each algorithm is measured under Wilcoxon's signed rank test with the significant level $\alpha = 0.05$.

| Function | BA | GA | PSO | SCA | WOA | CBH |
|---|---|---|---|---|---|---|
| $f_1$ | 1.39E+03 < | 5.30E+04 < | -1.40E+03 < | 4.18E+03 < | -1.28E+03 < | -1.28E+03 |
| $f_2$ | 2.13E+06 > | 8.30E+08 < | 9.49E+05 > | 7.66E+07 < | 4.23E+07 < | 4.37E+06 |
| $f_3$ | 2.20E+08 > | 6.83E+21 < | 3.10E+08 = | 2.61E+10 < | 1.11E+12 < | 9.46E+08 |
| $f_4$ | 6.18E+04 < | 1.18E+07 < | 1.16E+04 > | 2.89E+04 > | 7.87E+04 < | 3.77E+04 |
| $f_5$ | -9.99E+02 < | 3.72E+04 < | -1.00E+03 < | 3.10E+02 < | -7.11E+02 < | -1.00E+03 |
| $f_6$ | -8.71E+02 > | 1.67E+04 < | -8.76E+02 > | -3.14E+02 < | -7.75E+02 < | -8.67E+02 |
| $f_7$ | 2.88E+06 < | 3.38E+07 < | 4.10E+03 = | -5.62E+02 = | 1.45E+04 < | 4.63E+02 |
| $f_8$ | -6.79E+02 < | -6.79E+02 < | -6.79E+02 = | -6.79E+02 < | -6.79E+02 < | -6.79E+02 |
| $f_9$ | -5.76E+02 = | -5.71E+02 < | -5.77E+02 = | -5.75E+02 < | -5.76E+02 = | -5.77E+02 |
| $f_{10}$ | -4.98E+02 > | 6.20E+03 < | -4.98E+02 > | 3.07E+02 < | -2.86E+02 < | -4.98E+02 |
| $f_{11}$ | 1.76E+01 < | 4.26E+02 < | -1.47E+02 < | -1.93E+02 < | -9.20E+01 < | -3.80E+02 |
| $f_{12}$ | 3.41E+02 < | 7.35E+02 < | 2.05E+01 < | -6.26E+01 = | -9.83E+00 < | -7.48E+01 |
| $f_{13}$ | 4.58E+02 < | 7.90E+02 < | 2.11E+02 < | 3.41E+01 > | 7.59E+01 < | 5.06E+01 |
| $f_{14}$ | 3.28E+03 < | 6.09E+03 < | 2.75E+03 < | 4.23E+03 < | 3.36E+03 < | 2.61E+02 |
| $f_{15}$ | 3.27E+03 < | 5.58E+03 < | 2.71E+03 = | 4.58E+03 < | 3.63E+03 < | 2.60E+03 |
| $f_{16}$ | 2.02E+02 < | 2.04E+02 < | 2.01E+02 < | 2.02E+02 < | 2.02E+02 < | 2.01E+02 |
| $f_{17}$ | 1.09E+03 < | 1.76E+03 < | 5.61E+02 < | 5.89E+02 < | 6.54E+02 < | 3.40E+02 |
| $f_{18}$ | 1.20E+03 < | 1.86E+03 < | 6.46E+02 > | 6.89E+02 = | 7.60E+02 < | 7.23E+02 |
| $f_{19}$ | 5.25E+02 < | 3.72E+06 < | 5.18E+02 < | 1.47E+03 < | 5.55E+02 < | 5.04E+02 |
| $f_{20}$ | 6.10E+02 < | 6.10E+02 < | 6.10E+02 < | 6.10E+02 = | 6.10E+02 < | 6.10E+02 |
| $f_{21}$ | 1.03E+03 = | 4.37E+03 < | 1.04E+03 > | 1.73E+03 < | 1.35E+03 < | 1.05E+03 |
| $f_{22}$ | 5.07E+03 < | 7.41E+03 < | 4.53E+03 < | 5.34E+03 < | 4.79E+03 < | 1.37E+03 |
| $f_{23}$ | 5.11E+03 < | 7.14E+03 < | 4.85E+03 = | 5.86E+03 < | 5.15E+03 < | 4.86E+03 |
| $f_{24}$ | 1.31E+03 < | 1.41E+03 < | 1.29E+03 < | 1.28E+03 = | 1.28E+03 = | 1.27E+03 |
| $f_{25}$ | 1.36E+03 > | 1.45E+03 < | 1.41E+03 < | 1.38E+03 < | 1.38E+03 < | 1.36E+03 |
| $f_{26}$ | 1.51E+03 < | 1.61E+03 < | 1.52E+03 < | 1.41E+03 < | 1.49E+03 < | 1.40E+03 |
| $f_{27}$ | 2.37E+03 < | 2.69E+03 < | 2.40E+03 < | 2.29E+03 = | 2.26E+03 < | 2.26E+03 |
| $f_{28}$ | 5.94E+03 < | 8.85E+03 < | 4.82E+03 = | 4.07E+03 > | 5.86E+03 < | 4.98E+03 |
| $< / = / >$ | 21/2/5 | 28/0/0 | 15/7/6 | 19/6/3 | 25/3/0 | – |

It can be seen from the data in Table 3 that CBH has better performance on 21 functions than BA, similar performance on 2 functions, and poor per-

formance on 5 functions. CBH performs better on all functions than the traditional GA algorithm. CBH algorithm achieves better results than PSO algorithm on 15 functions, and among the remaining 13 functions, CBH performs almost the same as PSO on 7 functions, but do not perform well on the remaining 6 functions, especially on the $f_{18}$ and $f_{28}$ functions. Compared with the SCA algorithm, CBH performs worse than SCA on only three functions, $f_4$, $f_{14}$, and $f_{28}$ respectively. Compared with the WOA algorithm, CBH achieves better results on 25 functions, and the performance of the remaining 3 functions is similar.

Next, we select several function images to evaluate the convergence ability and the ability to jump out of the local optimum of CBH, BA, BH, GA, PSO, SCA, and WOA algorithms. The results are shown in Figure 6.




(a) $f_8$ (b) $f_{11}$ (c) $f_{12}$

(d) $f_{14}$ (e) $f_{15}$ (f) $f_{16}$

(g) $f_{17}$ (h) $f_{22}$ (i) $f_{23}$

(j) $f_{24}$ (k) $f_{26}$ (l) $f_{27}$

**Fig. 6** Comparision results of convergence performance of commom optimization algorirthms.

Overall, CBH can achieve the best performance on 15 functions when compared with the whole five algorithms. Here, we select 12 pictures with obvious effect to compare the convergence ability of the algorithm with the ability to jump out of the local optimum. It can be seen that although the convergence ability of CBH is not very competitive compared with other algorithms, CBH has a better ability to jump out of local optimization, especially on $f_8$, $f_{11}$, $f_{14}$, $f_{15}$, $f_{17}$, $f_{22}$ functions. Thererfore, we can conclude that CBH performs well in unimodal, multimodal, and composition functions. When other algorithms fall into local optimum, the CBH algorithm can jump out of the local optimum, gradually converge, and eventually tend to the global optimum.

### 4.4 Comparision between MCL algorithm and IMCL algorithm

Theoretically, using more hops to locate unknown mobile nodes can further improve the positioning accuracy, but the MCL algorithm only uses anchor nodes one and two hops away from the unknown node to locate it. In this paper, the MCL algorithm is improved, and more available anchor nodes are used to locate unknown nodes to further improve the positioning accuracy. Here, we compare the performance of the traditional MCL algorithm and IMCL algorithm to illustrate that the improvement of the Monte Carlo algorithm is necessary and effective.

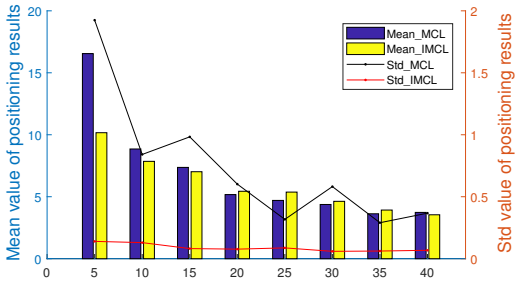



**Fig. 7** Comparison of MCL and IMCL performance in 100 m × 100 m deployment area with mobile nodes.

In the simulation experiment of this section, the total number of nodes is 200. We assume that the communication radius $R$ is 50, the maximum speed of node movement $Vmax$ is 20, and the deployment area is a square with side length $L$.

This section tests the performance of MCL and IMCL algorithms in deployment areas with side lengths of 100 m and 200 m, respectively, where the number of anchor nodes ranges from 5 to 40. Figure 7 and Figure 8 show the mean and standard deviation of MCL and IMCL algorithms after 20 rounds of tests. As can be seen from Figure 7, with the increase of the number of anchor nodes, the mean and standard deviation of the MCL algorithm decrease

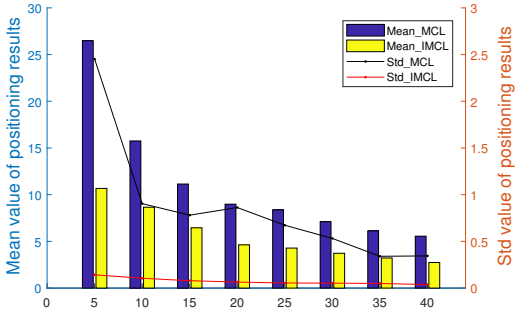



**Fig. 8** Comparison of MCL and IMCL performance in 200 m × 200 m deployment area with mobile nodes.

rapidly, and the IMCL algorithm also shows a downward trend on the whole. The standard deviation of IMCL is always far smaller than that of MCL, which shows that the stability of the IMCL algorithm is always better than that of the MCL. When the number of anchor nodes in the deployment area is 10, the mean of MCL is almost the same as that of IMCL, because the density of anchor nodes is already very large, it is difficult to further reduce the mean. Therefore, we extend the boundary length of the deployment area to 200 m. In Figure 8, the mean and standard deviation of MCL and IMCL decrease with the increase of the number of anchor nodes, but the performance of the IMCL algorithm is always much better than MCL, which shows that the IMCL algorithm can achieve better performance than MCL in scenarios with small anchor node density or relatively large deployment area. The figures above show that our improvement to the MCL algorithm is more effective.

## 5 Application in Monte Carlo localization

To verify that the CBH proposed in Section 3.1 can further improve the performance of IMCL, the simulation test carried out in this section applies several optimization algorithms such as CBH, Comprehensive Learning Particle Swarm Optimization (CLPSO) [19], Differential Evolution (DE) [27], Grey Wolf Optimization (GWO) [16, 23], PSO, WOA, and Adaptive Particle Swarm Optimization (APSO) [39] to IMCL algorithm to compare the performance of different algorithms.

For population-based algorithms, such as CLPSO, DE, GWO, PSO, WOA, and APSO, the population size $N_p$ is 30, but for CBH, the virtual population size is 300. The particle represents the possible position of the unknown node, and it continues to move towards the real position of the unknown node through iteration. Since the two dimensions of the initial value are different, the particle position update formula needs to be modified. Here, $x_\mu$ and $x_\sigma$ are used to update the X-axis value, $y_\mu$ and $y_\sigma$ are used to update the Y-axis value. The *winner* and the *loser* are still obtained by comparing the two

particles, as shown below, where $t$ represents the number of iterations.

$$x_{\mu}^{t+1} = x_{\mu}^{t} + \frac{1}{N_p}\left( winner(1) - loser(1)\right) \tag{14}$$

$$y_{\mu}^{t+1} = y_{\mu}^{t} + \frac{1}{N_p}\left( winner(2) - loser(2)\right) \tag{15}$$

$$x_{\sigma}^{t+1} = \sqrt{\left(x_{\sigma}^{t}\right)^2 + \left(x_{\mu}^{t}\right)^2 - \left(x_{\mu}^{t+1}\right)^2 + \frac{1}{N_p}\left(winner(1)^2 - loser(1)^2\right)} \tag{16}$$

$$y_{\sigma}^{t+1} = \sqrt{\left(y_{\sigma}^{t}\right)^2 + \left(y_{\mu}^{t}\right)^2 - \left(y_{\mu}^{t+1}\right)^2 + \frac{1}{N_p}\left(winner(2)^2 - loser(2)^2\right)} \tag{17}$$

Six groups of experiments are carried out in this section, and the deployment area is 200 m × 200 m. It can be seen from the data in Figure 9 and Table 4, when the number of anchor nodes reaches 30 in the 200 m × 200 m deployment area, the difference between various algorithms is very slight. In order to highlight the difference, the following experiment increases the number of nodes to 30 at most, and the mean and standard deviation of different algorithms are compared respectively. The experimental results are shown below:

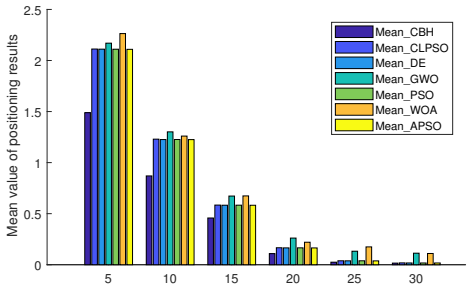



**Fig. 9** Comparison of CBH and common optimization algorithm performance in 200 m × 200 m deployment area with mobile nodes.

**Table 4** Comparison of standard deviation between CBH, CLPSO, DE, GWO, PSO, WOA, APSO algorithms in IMCL application, in which the deployment area is 200 m × 200 m.

| Nodes | CBH | CLPSO | DE | GWO | PSO | WOA | APSO |
|---|---|---|---|---|---|---|---|
| 5 | 5.516E-02 | 8.390E-02 | 8.487E-02 | 8.508E-02 | 8.459E-02 | 8.199E-02 | 8.453E-02 |
| 10 | 7.335E-02 | 7.980E-02 | 8.015E-02 | 8.102E-02 | 8.021E-02 | 8.238E-02 | 8.048E-02 |
| 15 | 5.631E-02 | 6.290E-02 | 6.274E-02 | 6.132E-02 | 6.261E-02 | 6.864E-02 | 6.287E-02 |
| 20 | 2.160E-02 | 2.549E-02 | 2.553E-02 | 2.723E-02 | 2.534E-02 | 2.189E-02 | 2.550E-02 |
| 25 | 8.932E-03 | 1.731E-02 | 1.749E-02 | 1.734E-02 | 1.722E-02 | 2.185E-02 | 1.733E-02 |
| 30 | 8.571E-03 | 9.217E-03 | 9.412E-03 | 1.012E-02 | 9.172E-03 | 1.402E-02 | 9.186E-03 |

In order to show the superiority of the CBH algorithm in the scenario of low anchor node density, the test deployment area continues to be expanded to 400 m × 400 m. The specific results are shown in the Figure 10.

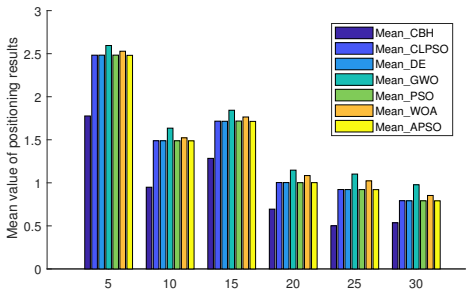



**Fig. 10** Comparison of CBH and common optimization algorithm performance in 400 m × 400 m deployment area with mobile nodes.

**Table 5** Comparison of standard deviation between CBH, CLPSO, DE, GWO, PSO, WOA, APSO algorithms in IMCL application, in which the deployment area is 400 m × 400 m.

| Nodes | CBH | CLPSO | DE | GWO | PSO | WOA | APSO |
|---|---|---|---|---|---|---|---|
| 5 | 1.187E-01 | 1.356E-01 | 1.350E-01 | 1.343E-01 | 1.361E-01 | 1.342E-01 | 1.355E-01 |
| 10 | 8.206E-02 | 1.117E-01 | 1.116E-01 | 1.127E-01 | 1.115E-01 | 1.130E-01 | 1.117E-01 |
| 15 | 1.111E-01 | 1.212E-01 | 1.191E-01 | 1.198E-01 | 1.204E-01 | 1.220E-01 | 1.197E-01 |
| 20 | 7.769E-02 | 8.233E-02 | 8.225E-02 | 7.844E-02 | 8.226E-02 | 7.940E-02 | 8.214E-02 |
| 25 | 8.570E-02 | 6.401E-02 | 6.310E-02 | 6.122E-02 | 6.332E-02 | 6.459E-02 | 6.339E-02 |
| 30 | 6.107E-02 | 7.799E-02 | 7.795E-02 | 7.551E-02 | 7.790E-02 | 7.666E-02 | 7.800E-02 |

In Figure 10, it can be seen more clearly that when the node deployment area increases, the gap between algorithms gradually becomes obvious, and the positioning accuracy of IMCL algorithm optimized by the CBH algorithm is always the best under the same conditions. In Table 5, except that the standard deviation is slightly larger when the number of anchor nodes is 25, the standard deviation of the CBH algorithm is always the smallest in other cases, which shows that our algorithm has relatively good stability.

This section uses CBH algorithm to optimize IMCL, and compares the results with several common optimization algorithms applied to IMCL. We have carried out several groups of simulation experiments, and the deployment area is from 100 m × 100 m to 200 m × 200 m, and finally, the deployment area is expanded to 400 m × 400 m. As the deployment area increases, the differences between different algorithms become obvious. In general, the CBH algorithm proposed in this paper has the best performance on IMCL and achieved satisfactory results, which shows that our work is useful.

## 6 Conclusion

In WSN, due to the limitations of node memory, processor performance and power consumption, it is very important to locate unknown nodes quickly and accurately, especially in WSN where nodes can move. We combine the BH algorithm with the compact strategy and propose the CBH algorithm. At the same time, the traditional MCL algorithm is improved by using multiple hops, and the ICML algorithm is proposed to make it more suitable for WSN with sparse anchor nodes. We combine the CBH algorithm with the IMCL algorithm and compare it with several other common algorithms applied in IMCL. The results of simulation experiments show that the work we have done has greatly improved the positioning accuracy without significantly increasing the network burden and hardware cost.

## Declarations

**Data Availability** The data used to support the findings of this study are available from the corresponding author upon request.
**Conflict of interest** The authors declare that they have no conflict of interest.
**Funding** This research received no external funding.

## References

1. Arjunan, S., Sujatha, P.: Lifetime maximization of wireless sensor network using fuzzy based unequal clustering and aco based routing hybrid protocol. Applied Intelligence **48**(8), 2229–2246 (2018)
2. Chai, Q.w., Chu, S.C., Pan, J.S., Hu, P., Zheng, W.m.: A parallel woa with two communication strategies applied in dv-hop localization method. EURASIP Journal on Wireless Communications and Networking **2020**(1), 1–10 (2020)
3. Chai, Q.W., Chu, S.C., Pan, J.S., Zheng, W.M.: Applying adaptive and self assessment fish migration optimization on localization of wireless sensor network on 3-d te rrain. J. Inf. Hiding Multim. Signal Process. **11**(2), 90–102 (2020)
4. Chai, Q.W., Zheng, J.W.: Rotated black hole: A new heuristic optimization for reducing localization error of wsn in 3d terrain. Wireless Communications and Mobile Computing **2021** (2021)
5. Chen, H., Huang, P., Martins, M., So, H.C., Sezaki, K.: Novel centroid localization algorithm for three-dimensional wireless sensor networks. In: 2008 4th International Conference on Wireless Communications, Networking and Mobile Computing, pp. 1–4. IEEE (2008)
6. Cheng, C.T., Fallahi, K., Leung, H., Chi, K.T.: Cooperative path planner for uavs using aco algorithm with gaussian distribution functions. In: 2009 IEEE International Symposium on Circuits and Systems, pp. 173–176. IEEE (2009)

7. Chu, S.C., Tsai, P.W., Pan, J.S.: Cat swarm optimization. In: Pacific Rim international conference on artificial intelligence, pp. 854–858. Springer (2006)
8. Dellaert, F., Fox, D., Burgard, W., Thrun, S.: Monte carlo localization for mobile robots. In: Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C), vol. 2, pp. 1322–1328. IEEE (1999)
9. Enge, P.K.: The global positioning system: Signals, measurements, and performance. International Journal of Wireless Information Networks **1**(2), 83–105 (1994)
10. Foo, J.L., Knutzon, J., Kalivarapu, V., Oliver, J., Winer, E.: Path planning of unmanned aerial vehicles using b-splines and particle swarm optimization. Journal of aerospace computing, Information, and communication **6**(4), 271–290 (2009)
11. Gao, D., Wang, G.G., Pedrycz, W.: Solving fuzzy job-shop scheduling problem using de algorithm improved by a selection mechanism. IEEE Transactions on Fuzzy Systems **28**(12), 3265–3275 (2020)
12. Guldogan, M.B.: Consensus bernoulli filter for distributed detection and tracking using multi-static doppler shifts. IEEE Signal Processing Letters **21**(6), 672–676 (2014)
13. Gumaida, B.F., Luo, J.: A hybrid particle swarm optimization with a variable neighborhood search for the localization enhancement in wireless sensor networks. Applied Intelligence **49**(10), 3539–3557 (2019)
14. Gupta, G.P.: Improved cuckoo search-based clustering protocol for wireless sensor networks. Procedia Computer Science **125**, 234–240 (2018)
15. Hatamlou, A.: Black hole: A new heuristic optimization approach for data clustering. Information sciences **222**, 175–184 (2013)
16. Hu, P., Pan, J.S., Chu, S.C.: Improved binary grey wolf optimizer and its application for feature selection. Knowledge-Based Systems **195**, 105746 (2020)
17. Kulkarni, R.V., Venayagamoorthy, G.K.: Particle swarm optimization in wireless-sensor networks: A brief survey. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) **41**(2), 262–267 (2010)
18. Liang, J., Qu, B., Suganthan, P., Hernández-Díaz, A.G.: Problem definitions and evaluation criteria for the cec 2013 special session on real-parameter optimization. Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, Technical Report **201212**(34), 281–295 (2013)
19. Liang, J.J., Qin, A.K., Suganthan, P.N., Baskar, S.: Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE transactions on evolutionary computation **10**(3), 281–295 (2006)
20. Liu, K., Wang, S., Zhang, F., Hu, F., Xu, C.: Efficient localized localization algorithm for wireless sensor networks. In: The Fifth International Conference on Computer and Information Technology (CIT'05), pp. 517–523.

IEEE (2005)
21. Mirjalili, S.: Sca: a sine cosine algorithm for solving optimization problems. Knowledge-based systems **96**, 120–133 (2016)
22. Mirjalili, S., Lewis, A.: The whale optimization algorithm. Advances in engineering software **95**, 51–67 (2016)
23. Mirjalili, S., Saremi, S., Mirjalili, S.M., Coelho, L.d.S.: Multi-objective grey wolf optimizer: a novel algorithm for multi-criterion optimization. Expert Systems with Applications **47**, 106–119 (2016)
24. Niculescu, D., Nath, B.: Ad hoc positioning system (aps) using aoa. In: IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No. 03CH37428), vol. 3, pp. 1734–1743. Ieee (2003)
25. Patwari, N., Hero III, A.O.: Using proximity and quantized rss for sensor localization in wireless networks. In: Proceedings of the 2nd ACM international conference on Wireless sensor networks and applications, pp. 20–29 (2003). DOI 10.1145/941350.941354
26. Pehlivanoglu, Y.V., Baysal, O., Hacioglu, A.: Path planning for autonomous uav via vibrational genetic algorithm. Aircraft Engineering and Aerospace Technology **79**(4), 352–359 (2007)
27. Qin, A.K., Huang, V.L., Suganthan, P.N.: Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE transactions on Evolutionary Computation **13**(2), 398–417 (2008)
28. Savvides, A., Han, C.C., Strivastava, M.B.: Dynamic fine-grained localization in ad-hoc networks of sensors. In: Proceedings of the 7th annual international conference on Mobile computing and networking, pp. 166–179 (2001). DOI 10.1145/381677.381693
29. Sharma, V., Patel, R., Bhadauria, H., Prasad, D.: Policy for planned placement of sensor nodes in large scale wireless sensor network. KSII Transactions on Internet and Information Systems (TIIS) **10**(7), 3213–3230 (2016)
30. Sharmin, A., Anwar, F., Motakabber, S.: A noble approach of aco algorithm for wsn. In: 2018 7th International Conference on Computer and Communication Engineering (ICCCE), pp. 152–156. IEEE (2018)
31. Shu, J., Liu, L., Chen, Y., Hu, H.: A novel three-dimensional localization algorithm in wireless sensor networks. In: 2008 4th International Conference on Wireless Communications, Networking and Mobile Computing, pp. 1–3. IEEE (2008)
32. Singh, S., Mittal, E., et al.: Range based wireless sensor node localization using pso and bbo and its variants. In: 2013 International conference on communication systems and network technologies, pp. 309–315. IEEE (2013)
33. Song, P.C., Chu, S.C., Pan, J.S., Yang, H.: Phasmatodea population evolution algorithm and its application in length-changeable incremental extreme learning machine. In: 2020 2nd International Conference on Industrial Artificial Intelligence (IAI), pp. 1–5. IEEE (2020)
34. Thrun, S., Fox, D., Burgard, W., Dellaert, F.: Robust monte carlo localization for mobile robots. Artificial intelligence **128**(1-2), 99–141 (2001)

35. Wang, G.G., Tan, Y.: Improving metaheuristic algorithms with information feedback models. IEEE transactions on cybernetics **49**(2), 542–555 (2017)
36. Wu, B.F., Jen, C.L.: Particle-filter-based radio localization for mobile robots in the environments with low-density wlan aps. IEEE Transactions on Industrial Electronics **61**(12), 6860–6870 (2014)
37. Yang, X.S., Deb, S.: Cuckoo search via lévy flights. In: 2009 World congress on nature & biologically inspired computing (NaBIC), pp. 210–214. Ieee (2009)
38. Yang, X.S., Gandomi, A.H.: Bat algorithm: a novel approach for global engineering optimization. Engineering computations **29**(5), 464–483 (2012)
39. Zhan, Z.H., Zhang, J., Li, Y., Chung, H.S.H.: Adaptive particle swarm optimization. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **39**(6), 1362–1381 (2009)
40. Zhang, S., Yan, S., Hu, W., Wang, J., Guo, K.: A component-based localization algorithm for sparse sensor networks combining angle and distance information. KSII Transactions on Internet and Information Systems (TIIS) **9**(3), 1014–1034 (2015)
41. Zheng, S.j., Kai, L., Zheng, Z.: Three dimensional localization algorithm based on nectar source localization model in wireless sensor network. Application Research of Computers **25**(8), 2512–2513 (2008)
42. Zheng, W.M., Liu, N., Chai, Q.W., Chu, S.C.: A compact adaptive particle swarm optimization algorithm in the application of the mobile sensor localization. Wireless Communications and Mobile Computing **2021** (2021)