Hindawi
Wireless Communications and Mobile Computing
Volume 2020, Article ID 3845407, 17 pages
https://doi.org/10.1155/2020/3845407

Research Article

A Hybrid Mobile Node Localization Algorithm Based on Adaptive MCB-PSO Approach in Wireless Sensor Networks

Hua Wu,[1,2] Ju Liu,[1] Zheng Dong,[1] and Yang Liu[2,3]

[1]*School of Information Science & Engineering, Shandong University, Qingdao, China*
[2]*School of Information Science & Electric Engineering, Shandong Jiaotong University, Jinan, China*
[3]*CAAC Key laboratory of General Aviation Operation, Civil Aviation Management Institute of China, Beijing, China*

Correspondence should be addressed to Ju Liu; juliu@sdu.edu.cn

Received 29 September 2019; Revised 17 March 2020; Accepted 21 May 2020; Published 13 June 2020

Academic Editor: Reza Monir Vaghefi

In this paper, a hybrid adaptive MCB-PSO node localization algorithm is proposed for three-dimensional mobile wireless sensor networks (MWSNs), which considers the random mobility of both anchor and unknown nodes. An improved particle swarm optimization (PSO) approach is presented with Monte Carlo localization boxed (MCB) to locate mobile nodes. It solves the particle degeneracy problem that appeared in traditional MCB. In the proposed algorithm, a random waypoint model is incorporated to describe random movements of anchor and unknown nodes based on different time units. An adaptive anchor selection operator is designed to improve the performance of standard PSO for each particle based on time units and generations, to maintain the searching ability in the last few time units and particle generations. The objective function of standard PSO is then reformed to make it obtain a better rate of convergence and more accurate cost value for the global optimum position. Furthermore, the moving scope of each particle is constrained in a specified space to improve the searching efficiency as well as to save calculation time. Experiments are made in MATLAB software, and it is compared with DV-Hop, Centroid, MCL, and MCB. Three evaluation indexes are introduced, namely, normalized average localization error, average localization time, and localization rate. The simulation results show that the proposed algorithm works well in every situation with the highest localization accuracy, least time consumptions, and highest localization rates.

1. Introduction

Wireless sensor networks (WSNs) are a kind of new generated networks which are composed of a large number of smart devices, namely, smart sensors, having sensing, computing, and communicating units [1–3]. Nevertheless, the tiny smart sensors are autonomous that are embedded with limited power sources, i.e., onboard batteries, which are generally nonrenewable. Advances in wireless networks, communication, and integration have enabled this new generation of WSNs suitable for a range of commercial and military applications [4, 5]. In such applications, the data reported by the sensors is often insignificant in no conjunction with the accurate knowledge of the physical positions of nodes. Therefore, determining the locations of the sensors, often called localization, is an important problem in WSNs, not only in its own right but also often as the first step towards solving more complicated and diverse network tasks.

Equipping each sensor by a global positioning system (GPS) [6, 7] seems to be the first choice to locate the sensors, due to the high positioning accuracy of GPS in outdoor environments. However, this solution is impractical in WSNs, especially in indoor environments where GPS signals are not reliable. Moreover, GPS receivers are energy consuming, costly, and bulky for tiny sensors. Hence, to extend the lifetime of the network, hardware, communication protocols, and processing algorithms should all be designed in a way with low energy consumption [8]. Alternative solutions have been proposed in the literature, considering some sensors, called anchor nodes, having known positions, and the others, called unknown nodes, requiring to be localized. In practice, anchor node positions can be obtained by using GPS or by installing anchors at points with known coordinates [9].

Until now, most previous research has focused on exploiting static localization problems, in which both anchor nodes and unknown nodes cannot move once they are deployed. Many high efficient and accurate localization algorithms have been proposed [10–12]. Although some algorithms could be applied iteratively for mobile nodes, very few recent localization methods discuss the mobility issue of the sensor nodes specially. Mobile node localization is far more complicated than static ones. Localization schemes with high accurate positioning information cannot be implemented by mobile sensors since they usually require centralized processing that takes too much time to run. Centralized schemes also make assumptions about network topology, which is not applicable to mobile WSNs [13]. Mobility can affect the localization process in many ways. One of the prime concerns regarding mobile sensor networks is latency. Longer time taken by localization may cause latency, as the sensor will have changed its position since the measurement took place. Doppler shift [14] is another issue in mobile WSNs. Doppler shift can occur when the transmitter of a signal is moving relative to the receiver. Moreover, since most of the proposed localization techniques require LOS, the movements of mobile sensor nodes may cause the localization to take place in a degraded LOS position [15].

There are three main kinds of scenarios for mobile WSNs. One is that anchor nodes are moving but unknown nodes are static. Another is the opposite with unknown nodes moving but anchor nodes static. The third is that both unknown and anchor nodes are dynamic. In this paper, the last one is taken into account, which is more general. Some mobile localization algorithms have been proposed. Salari et al. use mobility information of sensor nodes to improve the accuracy of the localization algorithm. In another one [8], both fingerprint and accelerometer information are incorporated for localization, which is proven to be better in localization accuracy. An algorithm is proposed using basic directional and meandering mobility models for the localization of moving sensor nodes [16]. The particle filter-based algorithm uses a mobile robot to evaluate the performance of the algorithm. It measures and stores the RSS data continuously and applies the algorithm to track the position of mobile robots [17]. A pedestrian group detection and tracking algorithm has been presented, which develops a novel temporal-spatial method for grouping and an event detection technique for contextual behavior recognition [18]. Another important kind of mobile localization algorithms is the Monte Carlo localization (MCL) method [19, 20] and their related ones, such as the Sequential Monte Carlo method [21, 22] and Monte Carlo localization boxed (MCB) [23, 24]. These Monte Carlo-based methods are easily implemented and energy efficient with high accuracy, which become a hot research field.

This paper presents a hybrid adaptive MCB-PSO localization algorithm for mobile WSNs, in which the mobility of both anchor and unknown nodes is taken into account. An improved particle swarm optimization (PSO) approach is proposed with Monte Carlo localization boxed (MCB) to realize mobile node localization. It could solve particle degeneracy problems that usually appeared in traditional MCB. In this algorithm, a random waypoint model is incorporated to describe the random movements of anchor and unknown nodes in different time units. An adaptive anchor selection operator is generated for each particle based on different time units and generations, maintaining the searching ability in the last few time units and particle generations. The objective function is redesigned to obtain a better rate of convergence and more accurate cost values. Furthermore, the moving scope of each particle is constrained in a specified space to improve the searching efficiency as well as save calculation time. Experiments are made in MATLAB software, and it is compared with DV-Hop, Centroid, MCL, and MCB. The results prove that the proposed algorithm works well in every situation with the highest localization accuracy, least time consumptions, and high localization rates.

The remainder of this paper is organized as follows. The problem descriptions and backgrounds are provided in Section 2. How to realize the algorithm and the process in detail is given in Section 3. Section 4 presents the experimental results which are accomplished in MATLAB software. Finally, the paper is concluded in Section 5.

2. Problem Descriptions and Backgrounds

Problem descriptions are first given in this section. Then, the background knowledge of MCB and standard PSO is described briefly. The advantages and disadvantages of the two could also be found in the end.

2.1. Problem Description. Traditional static wireless sensor networks are usually considered as delay-tolerant [25, 26]. Due to the node mobility, a real-time component for the localization algorithms is introduced. In this way, it makes a sensor network delay intolerant: information gathering and location calculation should happen in a real-time manner, depending on the speeds of both the unknowns and the anchors. In a mobile wireless sensor network, algorithms relying on global knowledge such as the calculation of hop counts or distances to all the anchors in the network should be avoided. Similarly, a mobile node cannot really benefit from iterative localization techniques where the location estimation is refined whenever a node receives more information from the network [27]. Besides possible information decay, a localization algorithm deployed in a mobile wireless sensor network should be able to cope with the temporary lack of anchors.

In this paper, a more general scenario that both unknown and anchor nodes are moving randomly in three-dimensional space is addressed. It could be applied to any scenario where the unknown and anchor nodes are deployed in an ad hoc way, both of which move either because of the environment they are in (wind, currents, etc.) or because they have actuators for motion. A random waypoint mobility model [28] is incorporated. Continuous time is supposed to be divided into discrete time units, and there are M anchor nodes and N unknown nodes deployed in a specified network. All nodes will have the same communication range and maximum moving speed. Since an unknown node may move away from its previous location, it needs to be

relocalized in each time unit. As sensor nodes in the network, although positions in the previous time unit could be used in the current time unit, prior location information will become increasingly inaccurate. The objective of the algorithm proposed in this paper is to localize all the unknown nodes in each time unit with high accuracy, low energy cost, and high localization rate under the help of anchor nodes.

2.2. Background. Brief descriptions of Monte Carlo localization boxed (MCB) and standard particle swarm optimization (PSO) approach are given in this part.

2.2.1. Monte Carlo Localization Boxed (MCB). Monte Carlo localization boxed (MCB) is derived from Monte Carlo localization (MCL) [29], in which each unknown node will establish its own one-hop and two-hop anchor sets by listening in order to restrict impossible samples. To further constrain the area where the samples are drawn, in MCB, anchor set information is used to limit the sample space which saves a lot of energy for a sensor node and improves the accuracy of sampling. MCB overcomes the defects in drawing samples and is a long and tedious process [27]. By doing this, two consequences could be obtained. First, good samples are drawn more easily and is thus faster, which rejects samples less often in the filtering phase, thereby reducing the number of iterations the algorithm needs to fill the sample set entirely. The second is implementation dependency.

Figure 1 shows the core idea of the MCB algorithm. The method used for constraining the sampling area is as follows. Unknown node listening data packages form anchors nearby. Based on the received information, one-hop or two-hop anchor sets are generated. As given in Figure 1, there is a box that covers the region where the anchor's communication range R overlaps. In other words, this box is the region of the deployment area where the unknown node possibly localizes. Only three one-hop anchors are heard by an unknown node in Figure 1, and the anchor box is established. For the three anchors, unknown nodes generate a square with the anchor as the center and $2R$ as the side length. The process of anchor box establishment consists in calculating coordinates $(x_{\min}, x_{\max})$ and $(y_{\min}, y_{\max})$ as follows:

$$\begin{aligned} x_{\min} &= \max_j \left(x_j - R\right), \\ x_{\max} &= \min_j \left(x_j + R\right), \\ y_{\min} &= \max_j \left(y_j - R\right), \\ y_{\max} &= \min_j \left(y_j + R\right). \end{aligned} \tag{1}$$

In equation (1) (x_j, y_j) is the coordinate of the jth anchor node in one hop. Similarly, for other anchor nodes in two hops, replace R with $2R$ in equation (1).

After the anchor box is established, the unknown node should draw samples within the region that the anchor box covers. Since the anchor box is a bare approximation of the radio range of the anchors, a filtering step is still included in the original MCL. The prediction and filtering steps repeat

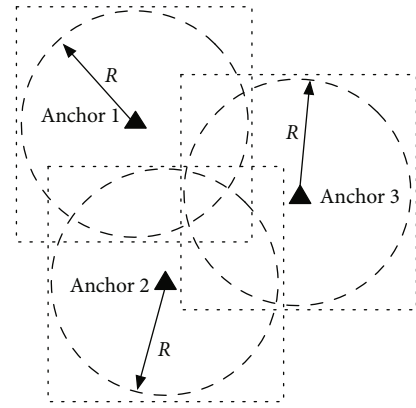


Figure 1: Anchor box establishment.

until the sample set is full or until the maximum number of tries is reached. Finally, the estimated position of the unknown node in the current time unit could be calculated.

However, there are also defects in MCB algorithms. Because all the candidate positions are generated randomly in an anchor box, no searching direction exists, which makes the time cost quite high and localization efficiency low. It may not be suitable for high mobility requirements. In extreme situations, most of the candidate nodes in the anchor box are not in the covered area of corresponding anchors when the iterations end, which makes the localization accuracy very low.

2.2.2. Standard Particle Swarm Optimization (PSO). Particle swarm optimization (PSO) is able to optimize the mathematical problem by iteratively trying to improve a candidate solution regarding a given measure of quality. It is a population-based stochastic computational technique that simulates the social behavior of a swarm of birds, flocking bees, and fish schooling. Its initialization begins randomly with random candidate solutions in searching space. A global optimum solution could be obtained by PSO in the end. The position and velocity are renewed iteratively in a given evolutionary system [30, 31]. In standard PSO, each particle is qualified with a certain dimension.

As given in equation (2) for any particle i, two crucial parameters are included: one is the particle position x_i and the other is the particle velocity v_i. The two will have the same dimension as D:

$$(\mathbf{x_i}, \mathbf{v_i}) = ((x_{i1}, x_{i2}, \cdots, x_{iD}), (v_{i1}, v_{i2}, \cdots, v_{iD})). \tag{2}$$

Swarm size stands for the total number of particles, which is written as S. In this way, the whole particle swarm can be denoted as follows in

$$\{(\mathbf{x_1}, \mathbf{v_1}), (\mathbf{x_2}, \mathbf{v_2}), \cdots, (\mathbf{x_S}, \mathbf{v_S})\}. \tag{3}$$

The objective function is indispensable in PSO, which is used to calculate the cost values for all the particles in the swarm in each iteration. Further, two kinds of cost values

```
Standard PSO algorithm

Set the generation counter k=0
/*Initialization*/ Generate S individuals x_i of D
dimensions with random initial location and speed in
searching space.
/*Mail loop*/
while termination criteria is not satisfied do
  generation counter k=k+1
  for i=1:S do
    /*Computation*/ Calculate cost value of particle x_i
using its current position in current generation.
    /*Update*/ Update the location and speed of particle
x_i of current generation based on formula (5).
    /*Update*/ Update local best position P_{i,best} and global
best position G_best based on current cost value.
  end for
end while
```

Figure 2: Pseudocode of standard PSO.

are stored, which are local best of each particle i and global best of all particles in the swarm. They can be denoted in

$$\begin{aligned}\mathbf{P_{i,best}} &= \left(p_{i1,\text{best}}, p_{i2,\text{best}}, \cdots, p_{iD,\text{best}}\right),\\ \mathbf{G_{best}} &= \left(g_{1,\text{best}}, g_{2,\text{best}}, \cdots, g_{D,\text{best}}\right).\end{aligned} \tag{4}$$

As analyzed above, there will be definitely S local best cost values in a swarm with S particles. But only one global best exists. After the two are obtained, positions and velocities of all particles in each dimension could be then updated by tracking the two best positions, local best and global best, using the following equation:

$$\begin{cases} v_{ij}^{k+1} = \mathrm{w}v_{ij}^{k} + c_1\zeta\left(p_{ij,\text{best}}^{k} - x_{ij}^{k}\right) + c_2\eta\left(g_{j,\text{best}}^{k} - x_{ij}^{k}\right), \\ x_{ij}^{k+1} = x_{ij}^{k} + \mathrm{r}v_{ij}^{k+1}, \end{cases} \tag{5}$$
$$i = 1, 2, \cdots, S, j = 1, 2, \cdots, D, k = 1, 2, \cdots, N-1.$$

In equation (5), w is inertia weight, which determines the impact of the previous velocity of a particle on its current iteration. c_1 and c_2 are positive constants, named self-cognition and social knowledge. They stand for the abilities to learn from the particle itself and the effects from the whole particle swarm, respectively. What is more, ξ and η are two random numbers between 0 and 1, which are to increase the randomness of PSO for global optimization. And r is a constant factor that can be used to constrain the position updating rate. Also, N is the total iterations that controls the number of iterations that the algorithm runs. The standard PSO will not stop until the termination criteria are satisfied. The pseudocode of standard PSO in detail is given in Figure 2.

The advantages of standard PSO are that it could be easily implemented and fewer parameters are needed, as well as the information of the global best particle can be shared within a particle swarm during the iterations. It is good at solving a global complicated optimization problem. However, there are some internal problems for standard PSO. Firstly, it easily falls into premature and local optimum. Secondly, the searching ability in the latter iterations deteriorates sharply, which leads to low accuracy of convergence and solutions that cannot be obtained in some extreme situations.

3. Realization of the Proposed Algorithm

This section gives the descriptions of core problems needed to be addressed and the realization of the proposed models and algorithm in detail.

3.1. The Random Waypoint Mobility Model. In this paper, a random waypoint mobility model [28] is introduced to make all the sensor nodes move randomly in each time unit in the network. As supposed in the previous section, there are M anchor nodes and N unknown nodes deployed randomly in WSNs with the same maximum moving speed $V_{\max}$ at the initial time unit. Their coordinates in the kth time unit could be denoted as follows:

$$\begin{cases} A_j^k\left(a_j^k, b_j^k, c_j^k\right), & \mathrm{j} = 1, 2, \cdots, \mathrm{M}, \\ U_i^k\left(x_i^k, y_i^k, z_i^k\right), & \mathrm{i} = 1, 2, \cdots, \mathrm{N}. \end{cases} \tag{6}$$

The polar coordinate system is included to describe the locations of nodes between two consecutive time units. Combining with equation (6), the mobility model of unknown node i could be obtained in the following equation. Anchor node j will have a similar formation for the mobility in two consecutive time units:

$$\begin{cases} x_i^k = x_i^{k-1} + V_i \cdot \sin\theta \cdot \cos\alpha, \\ y_i^k = y_i^{k-1} + V_i \cdot \sin\theta \cdot \sin\alpha, \\ z_i^k = z_i^{k-1} + V_i \cdot \cos\theta, \end{cases} \tag{7}$$

As given in equation (7), V_i is a random variable of node moving speed between 0 and $V_{\max}$. θ is the angle between the direction of node movement and z axis, which is randomly determined between 0 and π. α is the angle between the projector of node movement on XOY plane and x axis, which is also generated randomly between 0 and 2π in each time unit. Obviously, θ and α are independent of each other. A boundary condition is incorporated to judge whether the sensor nodes move out of the localization space or not. After each random movement, the current position will be judged. Once any coordinate is out of the localization space, its new position in the current time unit would be regenerated by changing the plus sign in equation (7) into a minus sign. For example, if the y coordinate in the kth time unit is out of the localization space, it will be recalculated using $y_i^k = y_i^{k-1} - V_i \cdot \sin\theta \cdot \sin\alpha$. It is the same for x coordinate and z coordinate in the same situation. For simplicity in the proposed model, the time costed by nodes to move from one position in the current time unit to another in the next time unit is so little that could be neglected. It means this moving process could be accomplished instantly.

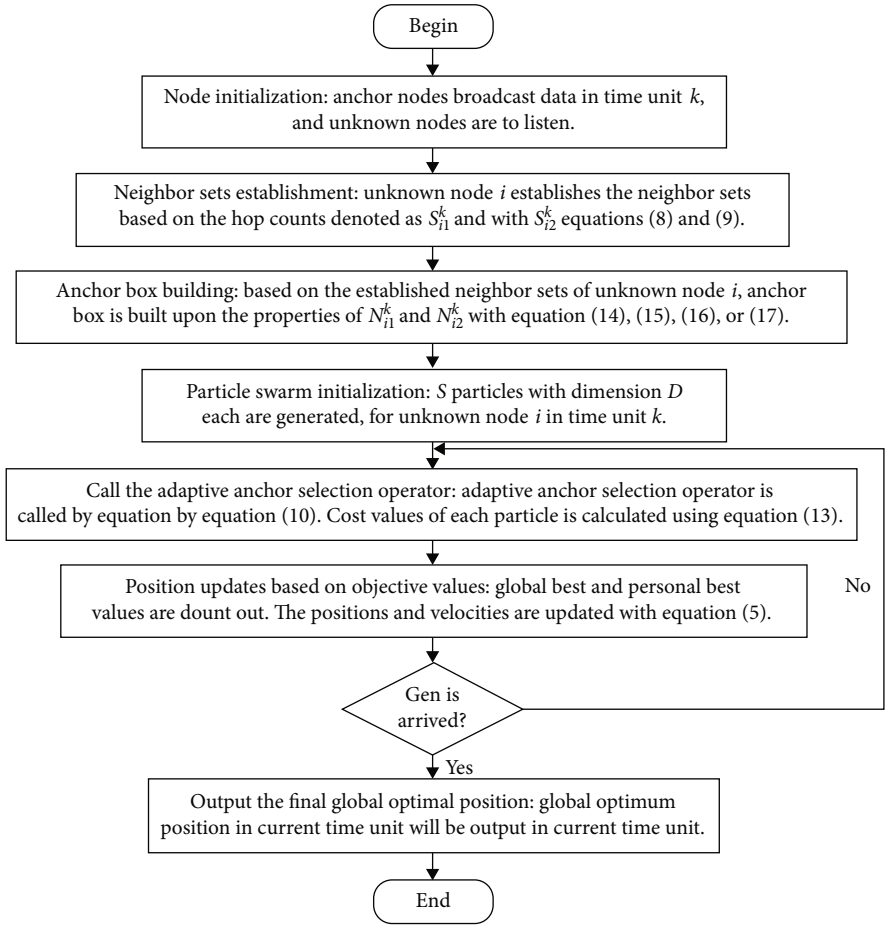


Figure 3: Flowchart of the proposed algorithm.

3.2. Adaptive Anchor Selection Operator. In the MCB algorithm, all the one-hop or two-hop anchors are incorporated in the sampling and filtering process to make the calculation of unknown nodes' coordinates more accurate. However, it makes the algorithm cost a great amount of time and energy, as well as fails to determine the positions of candidate particles in the specified anchor box in some extreme situations. Abundant anchor information may bring undesirable results. In order to solve this problem, an adaptive anchor selection operator is designed for particles in the anchor box used by the standard particle swarm optimization (PSO) approach when selecting anchor nodes.

In our proposed model, the adaptive anchor selection operator is based on the current time unit and particle generation. In the last few time units, error accumulations will be severe as well as serious particle degeneracy in the last few generations. For unknown node i at the kth ($k = 1, 2, \cdots, T$) time unit, there are N_{i1}^k one-hop and N_{i2}^k two-hop neighbor anchor nodes, which are formulated as two neighbor anchor sets in

$$\begin{aligned} S_{i1}^k &= \left\{A_{i1}^1, A_{i2}^1, A_{i3}^1, \cdots, A_{iN_{i1}^k}^1\right\}, \\ S_{i2}^k &= \left\{A_{i1}^2, A_{i2}^2, A_{i3}^2, \cdots, A_{iN_{i2}^k}^2\right\}. \end{aligned} \tag{8}$$

For simplicity, it can then be further reformed by the following equation:

$$\left\{S_{i1}^k, S_{i2}^k\right\} = \left\{A_{i1}^k, A_{i2}^k, \cdots, A_{i\left(N_{i1}^k + N_{i2}^k\right)}^k\right\}. \tag{9}$$

Suppose there are S particles generated randomly in node i's anchor box as a swarm. For any particle in the mth generation, it will decide the number of neighbor anchors which should be selected adaptively using the following equation:

$$N_{im}^k = \left\lfloor \left(\frac{k}{w_1 \sum k} + \frac{m}{w_2 \sum m}\right) \times \left(N_{i1}^k + N_{i2}^k\right) \right\rfloor. \tag{10}$$

N_{im}^k in equation (10) is the number of selected anchors for any particle in the mth generation in the anchor box of unknown node i in the kth time unit. $\lfloor \cdot \rfloor$ is an integer that is obtained by neglecting the decimal part. w_1 and w_2 are two weight coefficients, which are both set as 2 in our model. Once N_{im}^k is determined, every particle will select the anchors randomly from the neighbor anchor node set in equation (9), which will be needed in the objective function for global optimization.

3.3. Objective Function Design. An objective function is important in the global optimization problem. It decides

```
Start algorithm
For k=1:T
  Unknowns and anchors move as the model in equation (7) from ( k=1)th time unit.
  For i=1:N
    |*Node initialization *|
      In current time unit k anchor nodes broadcast data packages;
      Unknown node i listens to dat packages from anchor noes in the network;
    |*Neighbor sets establishment *|
      Unknown node i establishes the neighbor sets based on the hop counts using equation (8) and (9);
    |*Ancgor box building *|
      If N_{i1}^k = 0 && N_{i2}^k = 0
        C_i^A = { Null};
      else if N_{i1}^k ≠ 0 && N_{i2}^k = 0
        C_i^A = C_{i1}^1 ∩ C_{i2}^1 ∩⋯∩ C_{iN_{i1}^k}^1 ;
      else if N_{i1}^k = 0 and N_{i2}^k = 0
        C_i^A = C_{i1}^2 ∩ C_{i2}^2 ∩⋯∩ C_{iN_{i2}^k}^2 ;
      else if N_{i1}^k ≠ 0 and N_{i2}^k ≠ 0
        C_i^A = C_{i1}^1 ∩ C_{i2}^1 ∩⋯∩ C_{iN_{i1}^k}^1 ∩ C_{i1}^2 ∩ C_{i2}^2 ∩⋯∩ C_{iN_{i2}^k}^2 ;
      end if
      |*Particle swarm initialization *|
        S particles are generated with the same dimension D;
        Total interation is set as Gen.
        For m=1:Gen
          Call the adaptive anchor selection operator with equation (10);
          Calculated the cost values for each particle with equation (13);
      |*Particle position updating *|
          Base on the cost global best and personal best values are fount out;
          Positions of particles are updated with a certain velocity;
        end for
  end for
End for
```

Figure 4: Pseudocode of the proposed algorithm.

Table 1: Parameters of simulations.

Side length	Anchor deployment	Unknown node deployment	Number of time units	Movement in space	Performance index
100 m	Randomly	Randomly	29	Randomly	NALE NALT LR

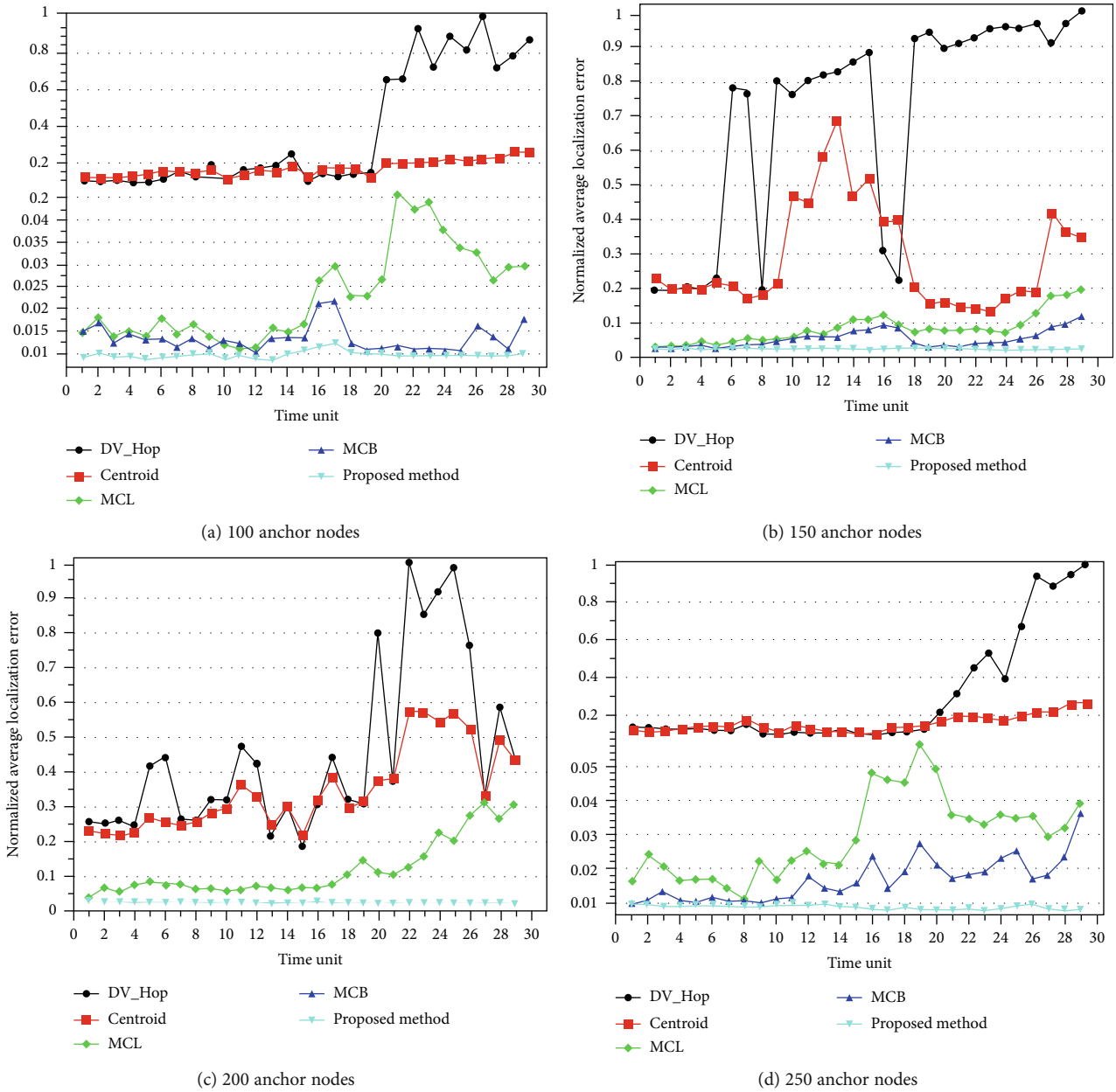
the searching abilities in the solution space, which also could evaluate the quality of candidate solutions. In this paper, a new objective function is redesigned for better performance. Many effective and important factors are taken into account.

3.3.1. Restricted Searching Space. As analyzed in the previous part, all the possible solutions lie in the formulated anchor box. In this way, each particle has to move inside of this specified anchor box space to find out the optimum solutions. But once it goes out of the box, it will be punished, which unfortunately brings a lot of localization error. For any particle j with position coordinate (x_j, y_j, z_j) and $j = 1, 2, \cdots, S$, the following equation could be obtained:

Table 2: Parameters for anchor node evaluation.

Items	Values	Items	Values
Number of anchor nodes	Variable	Particles in MCL & MCB	50
Number of unknown nodes	500	Samples in MCL & MCB	200
Communication range	25 m	Swarm of proposed method	250
Maximum moving speed	20 m/s	Particles of proposed method	3

$$J_1 = \begin{cases} \sum_{j=1}^{S} \left[\left| x_j - x_{\text{edge}} \right| + \left| y_j - y_{\text{edge}} \right| + \left| z_j - z_{\text{edge}} \right| \right], & \text{otherwise,} \\ 0, & \text{if particle inside of anchor box.} \end{cases} \tag{11}$$



(a) 100 anchor nodes

(b) 150 anchor nodes

(c) 200 anchor nodes

(d) 250 anchor nodes

Figure 5: NALE in each time unit with different numbers of anchors.

As described in equation (11) $(x_{\text{edge}}, y_{\text{edge}}, z_{\text{edge}})$ is the edge coordinate of unknown node i's anchor box.

3.3.2. Distances to Specified Anchors. Another parameter that should not be neglected is the distance to anchor nodes. Combining with equation (10), the number of anchor nodes for unknown node i in the mth generation at time unit kth could be determined, namely, N_{im}^{k}. Then, the estimated distance between particle j and each anchor node can then be calculated. The cost is denoted as the differences between estimated and true distances as in

$$J_2 = \sum_{l=1}^{N_{im}^{k}} \left[\left\| P_j - P_{A_{il}^{k}} \right\|_2 - D_{A_{il}^{k}} \right]. \tag{12}$$

As shown in equation (12), P_j is the position of particle j in the specified anchor box and $P_{A_{il}^{k}}$ is the position of the lth anchor given in equation (9). Also, $D_{A_{il}^{k}}$ is the measured distance to the lth anchor node. $\|\cdot\|_2$ denotes the Euclidean distance of any two positions.

Based on the two equations above, the final objective function for each particle can be written as follows:

$$J = J_1 + J_2. \tag{13}$$

3.4. Realization of the Proposed Algorithm. Once an adaptive anchor selection operator and the objective function are established, the whole proposed algorithm can be realized as follows.

Step 1 (node initialization). In current time unit k anchor, nodes broadcast their data packages to the whole network, including their IDs, positions, and hop counts. The unknown node i at time unit k is to listen to the data packages from anchor nodes from the network.

Step 2 (neighbor sets establishment). Each unknown node i at time unit k establishes the neighbor sets based on the hop counts. One-hop and two-hop sets are denoted as S_{i1}^k and S_{i2}^k, respectively, as in equation (8). Finally, all the neighbor anchors can be obtained using equation (9).

Step 3 (anchor box building). Based on the established neighbor sets of unknown node i, the anchor box is built in the following steps. Judge the properties of N_{i1}^k and N_{i2}^k first.

If $N_{i1}^k = 0$ and $N_{i2}^k = 0$, no neighbor anchor nodes exist for unknown node i. In this way, a virtual cube will be generated based on the node position in the (k-1)th time unit. The box will set the (k-1)th position as the center and $2 \cdot V_{\max}$ as the side length which is shown in

$$C_i^A = \{\text{Generation with } 2 \cdot V_{\max}\}. \tag{14}$$

If $N_{i1}^k \neq 0$ and $N_{i2}^k = 0$, each anchor node in set S_{i1}^k will build a cube with itself as center and $2R$ as the side length, denoted as C_{i1}^1. Suppose its number is N_{i1}^k. In this way, the anchor box would be built as

$$C_i^A = C_{i1}^1 \cap C_{i2}^1 \cap \cdots \cap C_{iN_{i1}^k}^1. \tag{15}$$

If $N_{i1}^k = 0$ and $N_{i2}^k \neq 0$, each anchor node in set S_{i2}^k will build a cube with itself as center and $4R$ as the side length, denoted as C_{i1}^2. Suppose its number is N_{i2}^k. And then, the anchor box would be built as

$$C_i^A = C_{i1}^2 \cap C_{i2}^2 \cap \cdots \cap C_{iN_{i2}^k}^2. \tag{16}$$

In the end, if $N_{i1}^k \neq 0$ and $N_{i2}^k \neq 0$, all the anchors in S_{i1}^k and S_{i2}^k will build cubes with themselves as centers and $2R$ and $4R$ as the side length, respectively. Their numbers N_{i1}^k are N_{i2}^k, respectively. The anchor box would be built as

$$C_i^A = C_{i1}^1 \cap C_{i2}^1 \cap \cdots \cap C_{iN_{i1}^k}^1 \cap C_{i1}^2 \cap C_{i2}^2 \cap \cdots \cap C_{iN_{i2}^k}^2. \tag{17}$$

Step 4 (particle swarm initialization). In our proposed algorithm, there will be S particles in total which are generated randomly for unknown node i in time unit k. Each particle has the same dimension D. The total iterations that each particle will go through are set as Gen.

Step 5 (call the adaptive anchor selection operator). Once the particle swarm is initialized for unknown node i, the adaptive anchor selection operator is called to determine which neighbor anchors will be used by equation (10). Then, the cost values of each particle are calculated using equation (13).

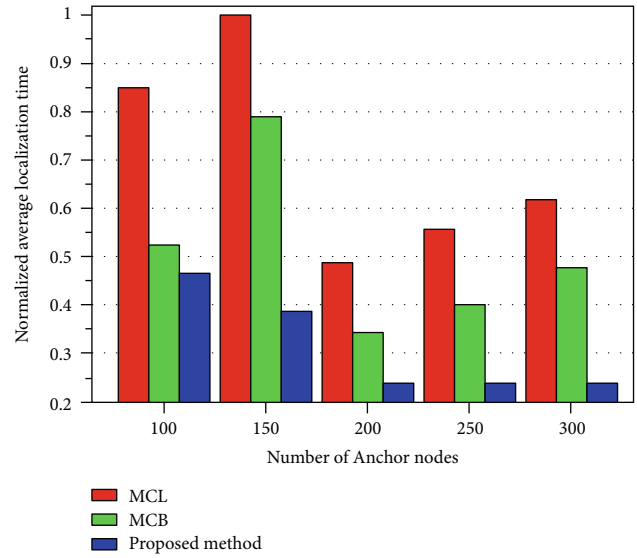


Figure 6: NALT with different numbers of anchors.

Table 3: Parameters for unknown node evaluation.

Items	Values	Items	Values
Number of anchor nodes	150	Particles in MCL & MCB	50
Number of unknown nodes	Variable	Samples in MCL & MCB	200
Communication range	25 m	Swarm of proposed method	250
Maximum moving speed	20 m/s	Particles of proposed method	3

Step 6 (position updates based on objective values). Based on the cost values of each particle, the global best and personal best values are found out. And then, the positions and velocities of the particles are updated using equation (5), respectively. Go to Step 5 until all the generations are satisfied.

Step 7 (output the final global optimal position). After the steps above are finished, global optimum position in the current time unit will be output in the current time unit.

The flowchart and pseudocode of the proposed algorithm are given in Figures 3 and 4.

4. Experiment Evaluation and Comparison

In this section, the proposed algorithm was tested and compared with DV_Hop, Centroid, MCL, and MCB. As mentioned before, DV_Hop and Centroid are two classic localization algorithms that could be used in mobile WSNs by position calculation iteratively in every time unit. They were implemented in MATLAB environment and ran on a server with a 2.8 GHz CPU and 16.0 GB of RAM.

4.1. Parameter Settings. We consider a three-dimensional space with a side length of 100 m, whose volume is 100 m ×

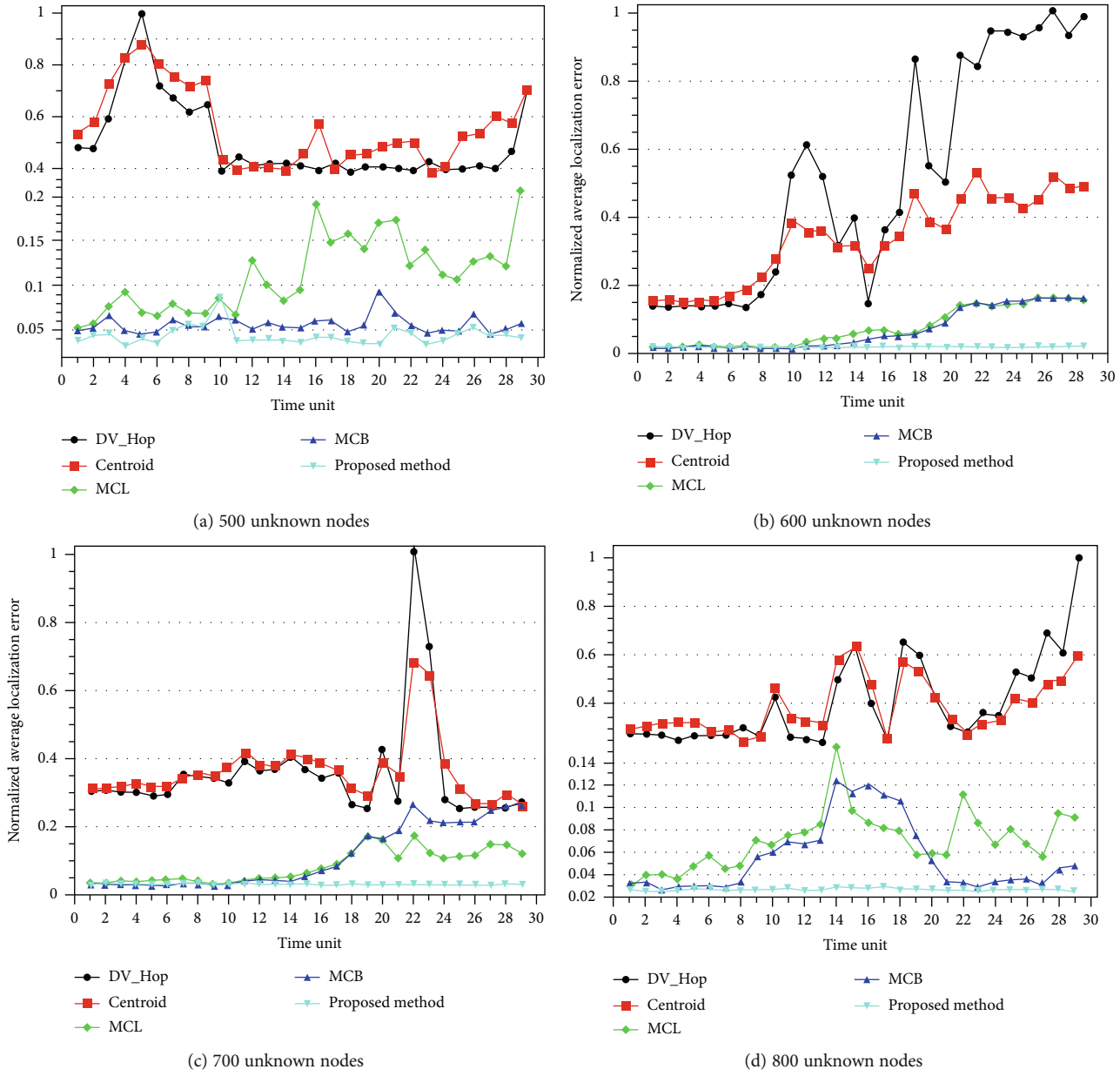


Figure 7: NALE in each time unit with different numbers of unknowns.

100 m × 100 m. For all our experiments, the anchor and unknown nodes are randomly distributed at the beginning. After the initialization, they begin to move with random directions and speed in the space. The moving speed is selected randomly between zero and maximum value. All the nodes share the same maximum moving speed. One time scope is comprised of 29 consecutive time units.

Three main performance indexes are introduced, normalized average localization error (NALE), normalized average localization time (NALT), and localization rate (LR). They are calculated in each time unit using the following equations:

$$\text{NALE} = \frac{1}{E_{\max} \cdot N} \sum_{i=1}^{N} \left\| P_i - \widehat{P}_i \right\|. \tag{18}$$

In equation (18), N is the number of unknown nodes that are deployed in the network. P_i and $\widehat{P}_i$ are the true and estimated positions of unknown node i. $\|\cdot\|$ denotes Euclidean distance. And $E_{\max}$ is the maximum average value of localization error of all unknown nodes in all the 29 time units:

$$\text{NALT} = \frac{1}{T_{\max} \cdot N} \sum_{i=1}^{N} t_i. \tag{19}$$

NALT here stands for localization time, which starts from the beginning of the algorithm to the end of the whole process. It includes the scenario establishment, parameter settings, calculation process, and position derivations. In equation (19), t_i is the time used by unknown node i to

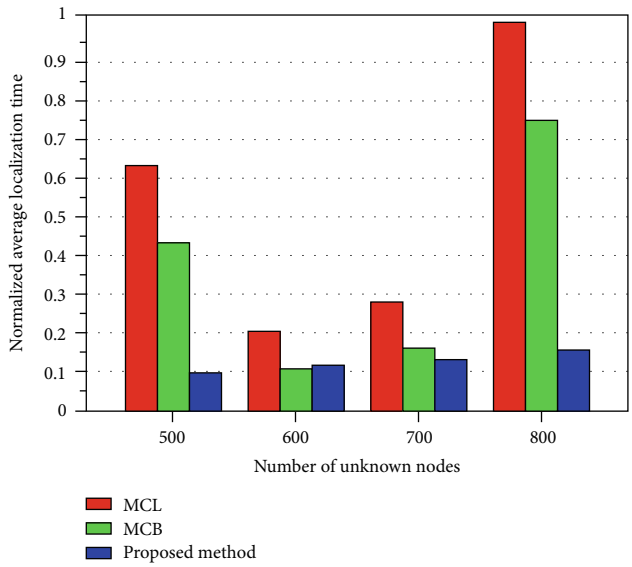


Figure 8: NALT with different numbers of unknown nodes.

realize location calculation. T_{max} is the longest average time needed in all the 29 time units.

The localization rate can be obtained with equation (20). In equation (20), M is the number of unknown nodes that could be localized in each time unit and N is the total number of unknown nodes:

$$\mathrm{LR} = \frac{M}{N}. \tag{20}$$

In order to better explore the performances, the effects of the number of anchors and unknown nodes, communication range, and maximum moving speed are compared separately. Swarm size is another parameter in our proposed algorithm. NALE and NALT under different swarm sizes are simulated at the end of this section. Parameter settings can be found in Table 1.

4.2. Effects of Anchor Nodes. In this section, the number of anchor nodes is considered to evaluate the performance of the proposed method. As given in Table 2, there are 500 sensor nodes that are needed to be localized. The number of anchors will be set as 100, 150, 200, and 250, respectively, in the simulation process. They share the same communication range 25 m and maximum moving speed 20 m/s. To better illustrate the performances, the number of particles and samples in MCL and MCB are the same as 50 and 200, respectively. For the proposed algorithm, two crucial parameters, namely, swarm size and particle size, are set as 250 and 3 as shown in Table 2.

Figure 5 presents the effects of anchor nodes on NALE. In this situation, the number of anchor nodes is set as 100, 150, 200, and 250 as in Figures 5(a)–5(d). It is obvious that DV_Hop and Centroid are the worst of all, no matter how many anchor nodes exist in the network. As the random movement of anchor and unknown nodes, the localization errors of the two are getting worse and worse in the later time units with a lot of fluctuations on their curves. That is because DV_Hop and Centroid were proposed to solve the static node localization problem in WSNs, which introduced a great number of errors when used in mobile WSNs. They are especially not suitable in the situations that all the sensor nodes moved randomly in 3D space. In Figure 5, MCL and MCB all present similar NALE trend. Especially in Figures 5(b)–5(d), the two show the same up and down trends. As time goes on, their NALE is getting larger and larger. In Figures 5(a), 5(b), and 5(d), MCB performs better than MCL in NALE. But when there are 200 anchor nodes as shown in Figure 5(c), MCL produces fewer errors than MCB after the 20th time unit.

Table 4: Parameters for communication range evaluation.

Items	Values	Items	Values
Number of anchor nodes	150	Particles in MCL & MCB	50
Number of unknown nodes	500	Samples in MCL & MCB	200
Communication range	Variable	Swarm of proposed method	250
Maximum moving speed	20 m/s	Particles of proposed method	3

There is no doubt that the proposed algorithm performs the best with the smallest errors in all the situations in Figure 5. Also, it is the most stable of the five. From the beginning time unit to the end, the proposed method can realize localization with the highest accuracy. Figure 5(c) shows the biggest differences among MCL, MCB, and the proposed method in the last time unit. It is 94.77% and 95.54% better than MCL and MCB, respectively.

MCL, MCB, and the proposed algorithm are designed specifically for mobile WSNs. In this way, only their NALTs are compared, in which DV_Hop and Centroid are not included. Figure 6 shows the NALT comparisons with different numbers of anchor nodes. Obviously, the proposed algorithm costs the least time compared with MCL and MCB. MCB is better than MCL in time consumption. As the number of anchor nodes increases, the time needed gets less and less for the proposed one. MCL costs the most time no matter how the number of anchor nodes changes. There are no direct connections between NALT and number of anchors for MCL and MCB. That is because the two belong to a kind of random localization algorithm, which is determined by characteristic of themselves.

4.3. Effects of Unknown Nodes. Here, the number of unknown nodes is used to test how the proposed algorithm works under different scales of networks compared with the others. It is obvious that fewer unknown nodes mean smaller scale of wireless sensor network and vice versa. A good localization algorithm should be suitable for all kinds of networks with high localization accuracy and little time cost. As given in Table 3, the number of unknown nodes becomes a variable. There are 150 anchor nodes in the network. Also, all the nodes are qualified with the same communication range 25 m and maximum moving speed 20 m/s. Other parameters

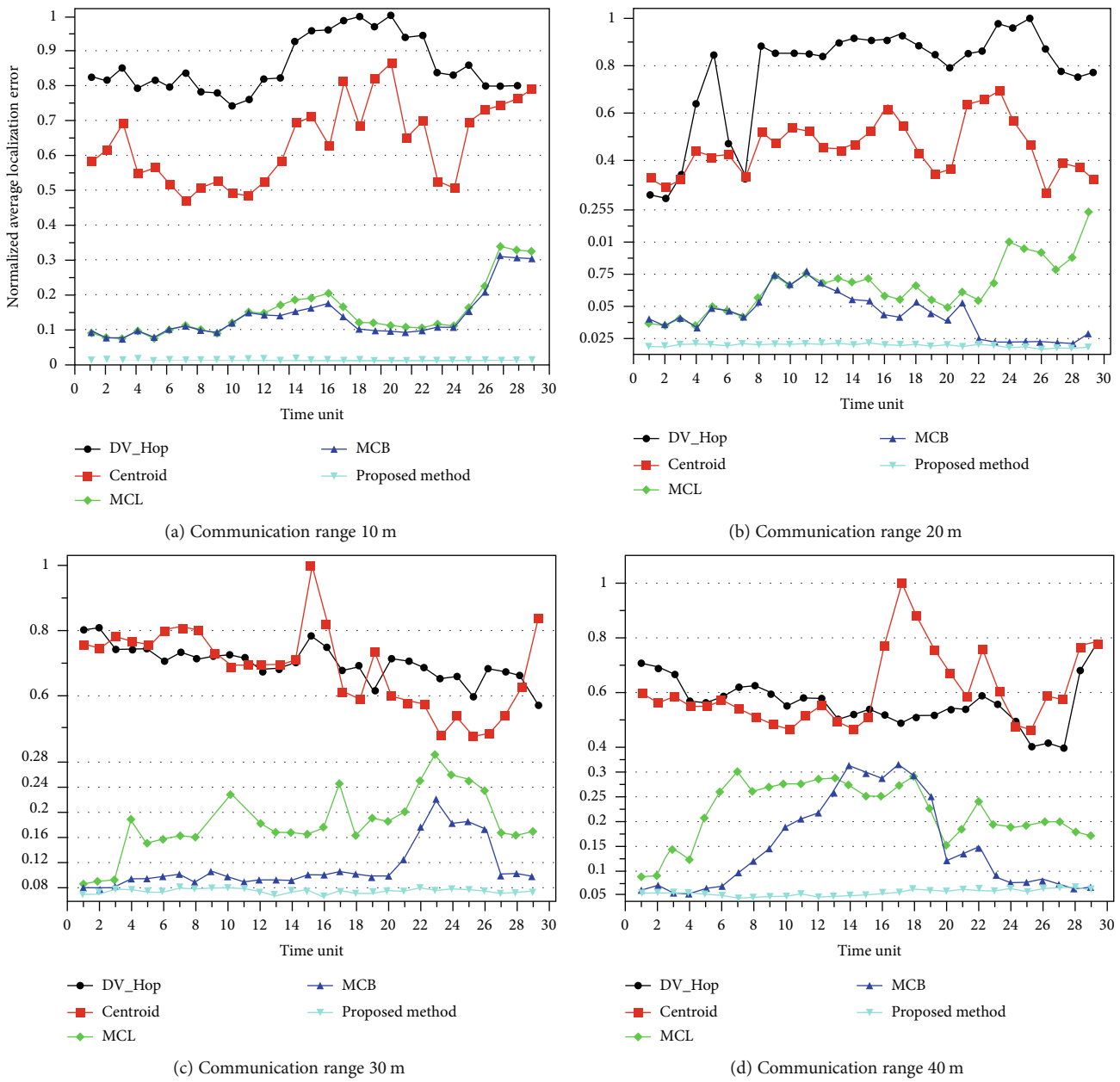


Figure 9: NALE in each time unit with different communication ranges.

are not changed as given in Table 2. Two indexes, NALE and NALT, are recorded, and the results are given in Figures 7 and 8.

As presented in Figure 7, DV_Hop and Centroid are still the worst of the five with big fluctuations. In Figures 7(a), 7(b), and 7(d), the differences between DV_Hop and Centroid and the other three are so big that two trends appear. When there are 700 and 800 sensor nodes needed to be localized, DV_Hop and Centroid will have the similar results. They show the same trend as the time unit goes on. But the accuracy is low compared with the other three. MCL and MCB obtain similar results in the first few time units. When 600 unknown nodes exist in the network, the curves of MCL and MCB almost coincide with each other. In Figure 7(c), in the last time unit, DV_Hop, Centroid, and MCB will have almost the same accuracy, in which point they almost coincide with each other. MCL outperforms MCB after the 20th time unit in Figure 7(c). But in Figure 7(a), MCB is better than MCL from the first time unit to the last time unit.

The proposed method performs the best with the fewest localization errors. Also, the curve of the proposed algorithm is the most stable except some little fluctuations in Figure 7(a). In Figure 7(b), after the 13th time unit, the differences among the proposed and MCL and MCB are getting more and more obvious. The trend is similar in Figures 7(c) and 7(d). After a specified time, our proposed method always outperforms MCL and MCB. That is because by using an improved PSO mechanism, the big randomness of node movement which gets worse and worse in the latter time units could be solved. But in MCL and MCB, the randomness

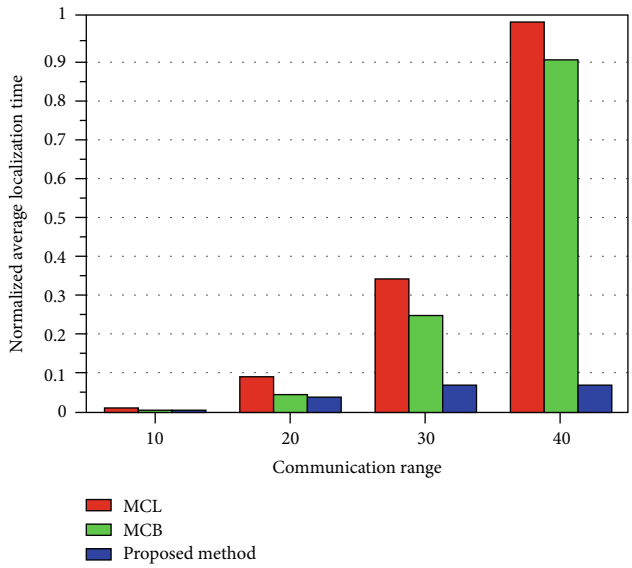


Figure 10: NALT with different communication ranges.

Table 5: Parameters for maximum moving speed evaluation.

Items	Values	Items	Values
Number of anchor nodes	150	Particles in MCL & MCB	50
Number of unknown nodes	500	Samples in MCL & MCB	200
Communication range	25 m	Swarm of proposed method	250
Maximum moving speed	Variable	Particles of proposed method	3

cannot be solved well, which correspondingly induces a lot of position errors for unknown nodes.

Figure 8 shows the time cost of MCL, MCB, and the proposed one. As the number of unknown nodes gets larger, correspondingly the proposed algorithm will cost more time. There are no relations between number of unknown nodes and localization time needed for MCL and MCB. But compared with MCL and MCB, our proposed is always the most efficient, which needs the least time no matter how the number of unknown nodes changes. The proposed algorithm is 85% and 79% better in time cost compared with MCL and MCB in the best situation, respectively.

4.4. Effects of Communication Range. In this paper, all the sensor nodes are supposed to be qualified with the same communication range. Once the sensor node enters the covered area of others, they could establish direct communication immediately. Table 4 provides explicit parameter settings. In this scenario, the number of anchor and unknown nodes is set as 150 and 500. They share the same maximum moving speed as 20 m/*s*.

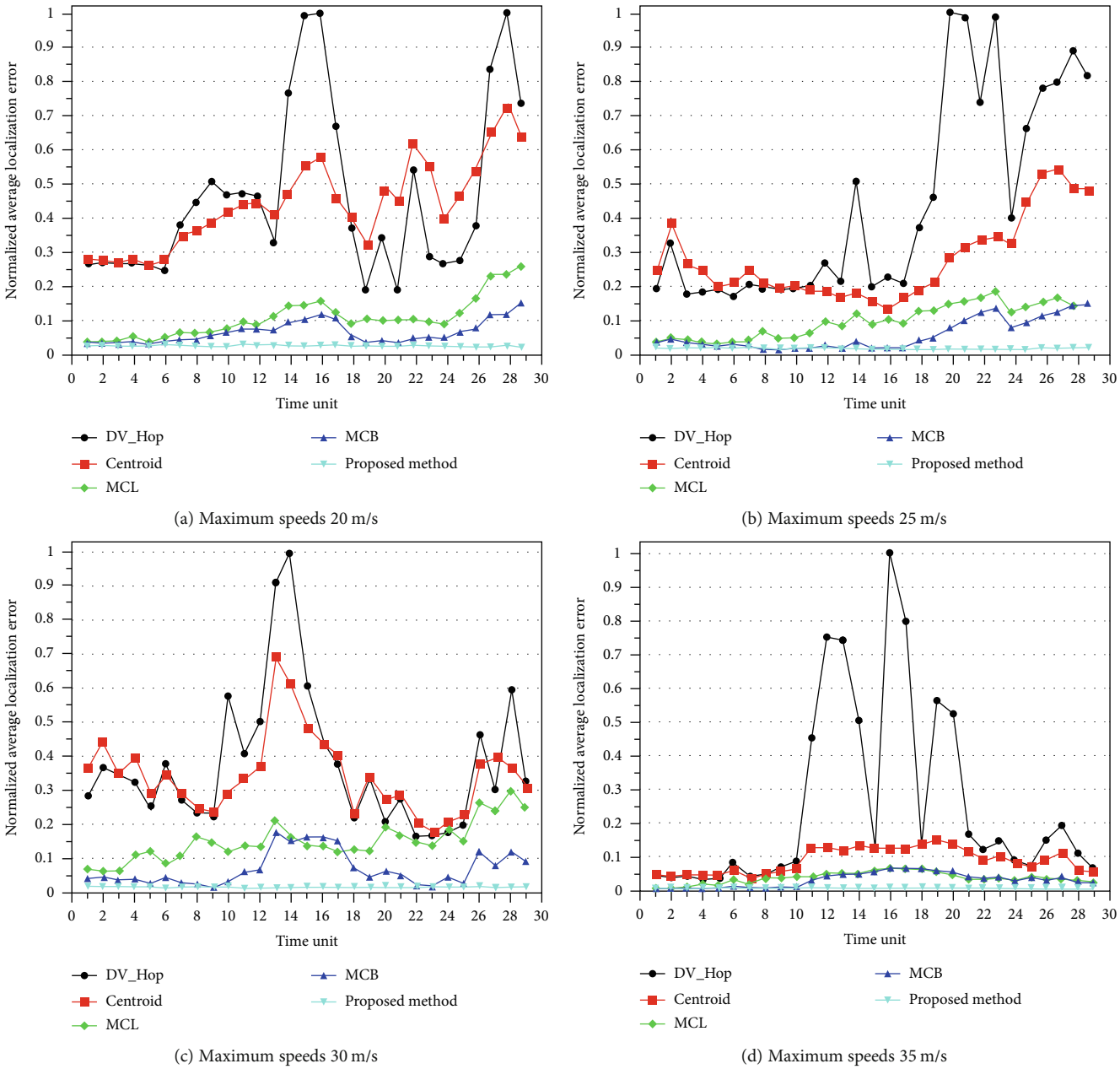
In Figure 9, DV_Hop and Centroid are the worst of all regardless of how the communication range is set. The two produce biggest NALE with biggest fluctuations. As in Figures 9(c) and 9(d), the differences between the two and the other three are the biggest. When the communication range is set as 10 m as in Figure 9(a), MCL and MCB will have the same change trend with similar localization accuracy. In Figure 9(b) with communication range 20 m, after the 12th time unit, the differences between MCL and MCB become more and more obvious. Unlikely in Figures 9(c) and 9(d), at the beginning, the two will generate big differences. From Figure 9, it can be seen that the localization error in (d) is larger than that in (b). It is because normalized average localization error is used for all the four figures in Figure 9. For NALE, firstly, the largest value in each subfigure is selected as the standard, and four local maximum errors will be found. Then, other values in each subfigure will be compared with maximum error. Finally, normalized value can be obtained using different standard values.

The proposed one in this paper is the best of all under all communication range settings. Especially when communication range is small, such as 10 m and 20 m, it still outperforms all the other algorithms with the highest localization accuracy. It is the most stable with least fluctuations from the first time unit to the end. The biggest improvement occurs with communication range 10 m in Figure 9(a); it is 97.19% and 97.01% better compared with MCL and MCB.

NALT with different communication range settings are given in Figure 10. All the three show an upward trend as the communication range gets larger and larger. That is because larger communication range means more neighbor nodes are taken into account during the localization process, which needs more processing time. Another from the bars in Figure 10, we can conclude that our proposed method is much better than MCL and MCB in time cost, which is more obvious with larger communication range. When the communication range is set as 40 m, the proposed will save about 93.83% and 93.16% localization time compared with MCL and MCB. In this way, the proposed one in this paper is more suitable for WSNs with high mobility and randomness, which obtains best localization accuracy and efficiency compared with others.

4.5. Effects of Maximum Moving Speed. Moving speed is important in mobile WSNs, which could destroy the topology of the networks and introduce a lot of randomness. Bigger moving speed means more randomness. It makes node localization get more complicated. In the model proposed in this paper, all the nodes will have the same maximum moving speed, whose final moving speed is selected randomly from zero and the maximum value. In the simulation process, there are 150 anchor nodes and 500 unknown nodes deployed randomly in the localization space. Their communication range is set as 25 m. The corresponding maximum speed is a variable which is 20 m/*s*, 25 m/s, 30 m/s and 35 m/s. Explicit parameter settings could be found in Table 5. Results of NALE and LR are provided in the end.

Figure 11 gives different NALEs in each time unit under different maximum moving speeds. As shown in Figures 11(a) and 11(b), the five curves could be divided into two groups based on the localization accuracy. The first is



(a) Maximum speeds 20 m/s

(b) Maximum speeds 25 m/s

(c) Maximum speeds 30 m/s

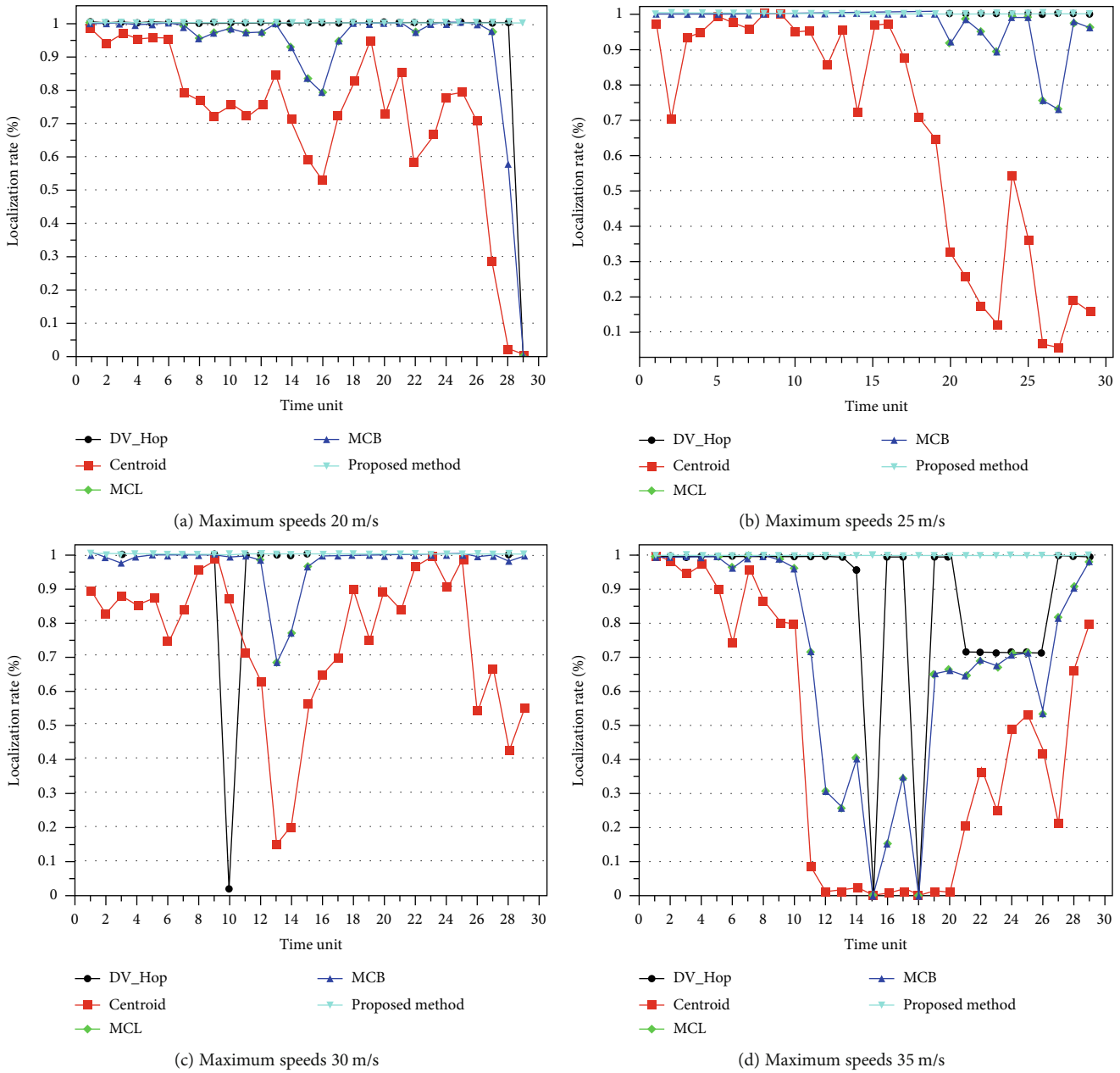
(d) Maximum speeds 35 m/s

Figure 11: NALE in each time unit with different maximum moving speeds.

DV_Hop and Centroid. The two produce largest NALE in every time unit. Big fluctuations are also introduced. Combining with Figures 11(c) and 11(d), larger maximum moving speed brings bigger fluctuation for DV_Hop. But for Centroid, when the maximum speed is at 35 m/*s*, the fluctuation is small. The second group is MCL, MCB, and the proposed one. With all the speed settings, MCL and MCB obtain bigger position errors as the time goes on. But for the curves of MCB in Figures 11(c) and 11(d), the peak value appears in the middle time units. When all nodes have the maximum moving speed 35 m/s, the differences between MCL and MCB are so small that could be neglected. As in Figure 11(d), the two curves are almost coinciding with each other. It is because when the maximum speed is too large, the improvements of MCB which is derived from MCL are so tiny that can produce limited effects on the localization accuracy.

There is no doubt that our proposed method is the best of all under any maximum moving speed settings. Also, it is the most stable of the four from the beginning to the ending time unit. No sharp fluctuations exist on the curves in all figures in Figure 11. It is much better than MCL and MCB, as well as DV_Hop and Centroid. From the data in Figure 11(a), we can conclude that it is 91.41% and 85.71% better than MCL and MCB in the last time unit.

In this section, another index is given, namely, localization rate (LR). It reflects the localization capacity of an algorithm. Comparisons between the five are made, and the



(a) Maximum speeds 20 m/s

(b) Maximum speeds 25 m/s

(c) Maximum speeds 30 m/s

(d) Maximum speeds 35 m/s

Figure 12: LR in each time unit with different maximum moving speeds.

results are shown in Figure 12. Centroid becomes the worst of all. Only when the maximum speed is set as 25 m/s, it can localize all the unknown nodes in the 9th and 10th time units. In other situations, it fails to realize all the unknown nodes. Moreover, its average LR of all time units under the four maximum speeds are 69.35%, 64.34%, 71.97%, and 43.61%, respectively. DV_Hop obtains the better LR, namely, 93.31%, 96.67%, 93.33%, and 84.16%. MCL and MCB have the same LR as shown in Figure 12. The values are 89.36%, 93.74%, 94.48%, and 66.99% correspondingly. It is not hard to find out that when the maximum moving speed is set as 35 m/s, all the four will generate the lowest localization rate compared with other maximum speed settings. It means bigger moving speed brings more randomness. But the proposed method could localize all the unknown nodes under every circumstance in the situation, which means high robustness and reliability.

NALT comparisons are given in Figure 13. The maximum moving speed of sensor nodes generates a great number of effects on MCL and MCB. Especially when the speed is set as 30 m/s, both MCL and MCB cost the most time. In all situations, MCB needs less time than MCL, which means MCB improves the localization efficiency of MCL largely. The proposed method is the most efficient with the least time cost under every scenario. No matter how the maximum speed changes, the time needed is stable, which means that the moving speed of sensor nodes will generate quite a few effects on the proposed method.

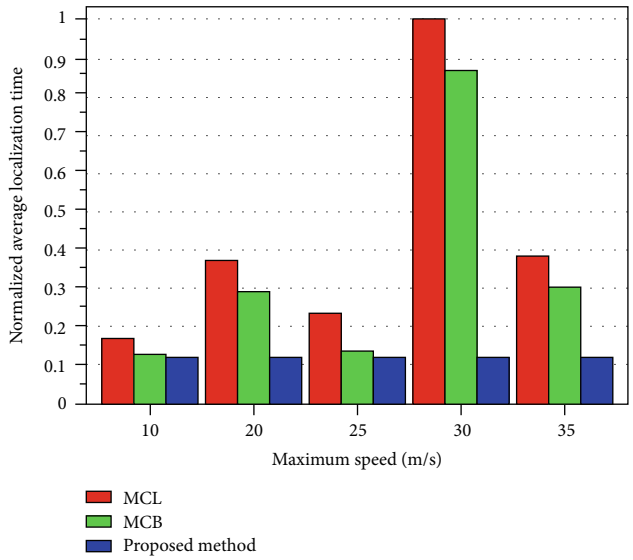


Figure 13: NALT with different maximum moving speeds.

Table 6: Parameters for different swarm size evaluation.

Items	Values	Items	Values
Number of anchor nodes	150	Maximum moving speed	20 m/s
Number of unknown nodes	500	Swarm of proposed method	Variable
Communication range	25 m	Particles of proposed method	3

4.6. Effects of Swarm Size. In order to further explore the proposed method, its performances under different swarm sizes are simulated in MATLAB software. Parameters in the simulation process are given in Table 6. The swarm size is set as 100, 200, 300, 400, 500 and 600, respectively.

Normalized average localization errors are listed in Figure 14. The six curves show a similar trend in NALE with similar fluctuations. When the swarm size is set as $S = 100$, the biggest NALE is generated. But when $S = 600$, the best performance can be obtained. As given in Figure 14, we can conclude that a larger swarm size in the proposed algorithm means a smaller NALE, namely, the best localization accuracy.

Normalized average localization time needed for the proposed algorithm is given in Figure 15. As the swarm size gets larger, more and more NALT is needed correspondingly. Combining with Figure 14, this parameter, namely, swarm size, should be selected carefully based on the different situations and requirements on NALT and NALE.

5. Conclusion

In this paper, a hybrid adaptive MCB-PSO node localization algorithm is proposed for mobile wireless sensor networks (MWSNs), which considers the mobility of both anchor and unknown nodes. An improved particle swarm optimization (PSO) approach is mixed with Monte Carlo localization

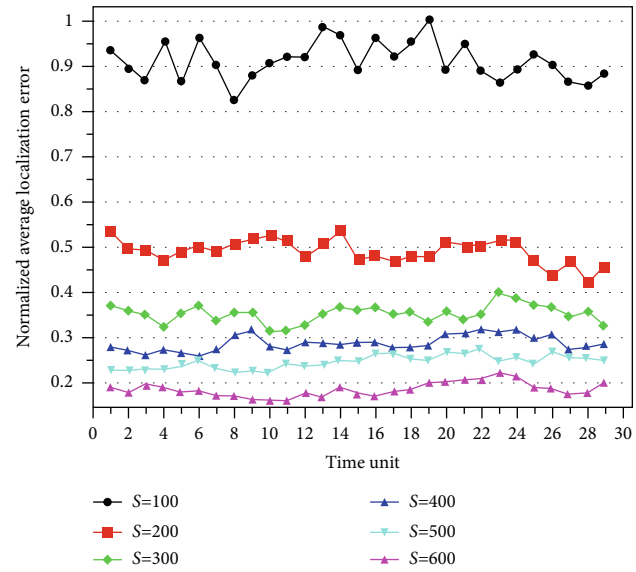


Figure 14: NALE with different swarm sizes of the proposed algorithm.

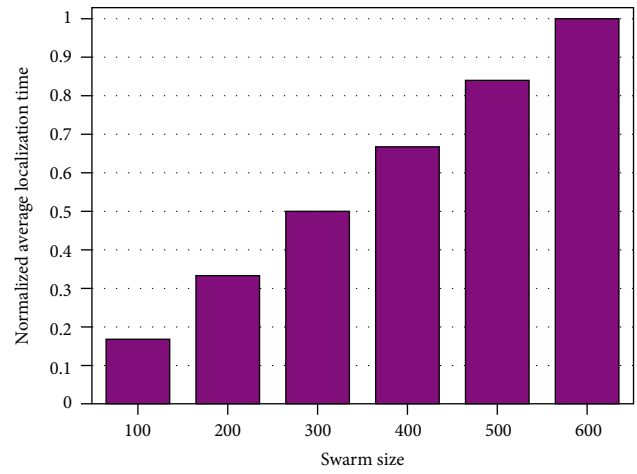


Figure 15: NALT with different swarm sizes of the proposed algorithm.

boxed (MCB) to realize mobile node localization as well as to solve the particle degeneracy problem that appeared in traditional MCB. In the proposed algorithm, a random waypoint model is incorporated to describe the movements of anchor and unknown nodes in each time unit. The objective function is redesigned to obtain a better rate of convergence and more accurate cost value. An adaptive anchor selection operator is specially designed for each particle based on different time units and generations, to maintain the searching ability in the last few time units and particle generations. Furthermore, the moving scope of each particle is constrained in a specified space to improve the searching efficiency as well as saving calculation time. Experiments are made in MATLAB software and compared with DV-Hop, Centroid, MCL, and MCB. The results prove that the proposed algorithm works well in every situation with the highest localization accuracy, least time consumptions, and biggest localization rate.

Data Availability

All the data used to support the findings of this study were supplied by Yang Liu under license and so cannot be made freely available. Requests for access to these data should be made to Yang Liu (ly0314@buaa.edu.cn).

Conflicts of Interest

The authors declare that they have no conflict of interests.

Acknowledgments

This work was funded and supported by the Research on the theory and method of aircraft autonomous safety separational maintenance in complex low-altitude dense flight (U1933130), CAAC Key laboratory of General Aviation Operation/Civil Aviation Management Institute of China (CAMICKFJJ-2019-01), Doctoral Research Initiation Fund of Shandong Jiaotong University (BS201902025), 1251 Talent Cultivation Project of Shandong Jiaotong University, project ZR201911010052 supported by Shandong Provincial Natural Science Foundation, and National Natural Science Funds of China (61901245).

References

[1] M. Ke, S. Tian, L. Lu, and C. Wang, "Robust power allocation for cooperative localization in jammed wireless sensor networks," *Wireless Communications and Mobile Computing*, vol. 2019, Article ID 6904687, 9 pages, 2019.

[2] Y. Zhou, L. Yang, L. Yang, and M. Ni, "Novel energy-efficient data gathering scheme exploiting spatial-temporal correlation for wireless sensor networks," *Wireless Communications and Mobile Computing*, vol. 2019, Article ID 4182563, 10 pages, 2019.

[3] H. A. Shah, I. Koo, and K. S. Kwak, "Actor–critic-algorithm-based accurate spectrum sensing and transmission framework and energy conservation in energy-constrained wireless sensor network-based cognitive radios," *Wireless Communications and Mobile Computing*, vol. 2019, Article ID 6051201, 12 pages, 2019.

[4] M. Zeng, X. Huang, B. Zheng, and X. Fan, "A heterogeneous energy wireless sensor network clustering protocol," *Wireless Communications and Mobile Computing*, vol. 2019, Article ID 7367281, 11 pages, 2019.

[5] F. Zhao and L. J. Guibas, *Wireless sensor networks: an information processing approach*, Morgan Kaufmann, 2004.

[6] R. S. Nerem and K. M. Larson, "Global positioning system, theory and practice, 5th edition," *Eos, Transactions American Geophysical Union*, vol. 82, no. 33, pp. 365–365, 2001.

[7] C. Savarese, J. M. Rabaey, and J. Beutel, "Location in distributed ad-hoc wireless sensor networks," in *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*, pp. 2037–2040, Salt Lake City, UT, USA, 2001.

[8] X. Lv, F. Mourad-Chehade, and H. Snoussi, "Decentralized localization using radio-fingerprints and accelerometer in WSNs," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 51, no. 1, pp. 242–257, 2015.

[9] S. Salari, S. Shahbazpanahi, and K. Ozdemir, "Mobility-aided wireless sensor network localization via semidefinite programming," *IEEE Transactions on Wireless Communications*, vol. 12, no. 12, pp. 5966–5978, 2013.

[10] Y. I. Wu, H. Wang, and X. Zheng, "WSN localization using RSS in three-dimensional space-a geometric method with closed-form solution," *IEEE Sensors Journal*, vol. 16, no. 11, pp. 4397–4404, 2016.

[11] Z. Farid, R. Nordin, M. Ismail, and N. F. Abdullah, "Hybrid indoor-based WLAN-WSN localization scheme for improving accuracy based on artificial neural network," *Mobile Information Systems*, vol. 2016, Article ID 6923931, 11 pages, 2016.

[12] B. Peng and L. Li, "An improved localization algorithm based on genetic algorithm in wireless sensor networks," *Cognitive Neurodynamics*, vol. 9, no. 2, pp. 249–256, 2015.

[13] I. Amundson and X. D. Koutsoukos, "A survey on localization for mobile wireless sensor networks," in *Mobile Entity Localization and Tracking in GPS-less Environnments. MELT 2009*, R. Fuller and X. D. Koutsoukos, Eds., vol. 5801 of Lecture Notes in Computer Science, pp. 235–254, Springer, Berlin, Heidelberg, 2009.

[14] M. B. Guldogan, "Consensus Bernoulli filter for distributed detection and tracking using multi-static Doppler shifts," *IEEE Signal Processing Letters*, vol. 21, no. 6, pp. 672–676, 2014.

[15] T. J. S. Chowdhury, C. Elkin, V. Devabhaktuni, D. B. Rawat, and J. Oluoch, "Advances on localization techniques for wireless sensor networks: a survey," *Computer Networks*, vol. 110, pp. 284–305, 2016.

[16] M. İ. Akbaş, M. Erol-Kantarcı, and D. Turgut, "Localization for wireless sensor and actor networks with meandering mobility," *IEEE Transactions on Computers*, vol. 64, no. 4, pp. 1015–1028, 2015.

[17] B. F. Wu and C. L. Jen, "Particle-filter-based radio localization for mobile robots in the environments with low-density WLAN APs," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 12, pp. 6860–6870, 2014.

[18] S. Li, Z. Qin, and H. Song, "A temporal-spatial method for group detection, locating and tracking," *IEEE Access*, vol. 4, pp. 4484–4494, 2016.

[19] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte Carlo localization for mobile robots," in *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*, Detroit, MI, USA, May 1999.

[20] S. Thrun, D. Fox, W. Burgard, and F. Dellaert, "Robust Monte Carlo localization for mobile robots," *Artificial Intelligence*, vol. 128, no. 1-2, pp. 99–141, 2001.

[21] A. Doucet, N. de Freitas, and N. Gordon, *Sequential Monte Carlo Methods in Practice*, Springer, 2001.

[22] J. E. Handschin, "Monte Carlo techniques for prediction and filtering of non-linear stochastic processes," *Automatica*, vol. 4, no. 6, pp. 555–563, 1970.

[23] F. L. Pedrocchi, N. E. Bonesteel, and D. P. DiVincenzo, "Monte Carlo studies of the self-correcting properties of the Majorana quantum error correction code under braiding," *Physical Review B*, vol. 92, no. 11, p. 115441, 2015.

[24] Z. Wang, Y. Wang, M. Ma, and J. Wu, "Efficient localization for mobile sensor networks based on constraint rules optimized Monte Carlo method," *Computer Networks*, vol. 57, no. 14, pp. 2788–2801, 2013.

[25] K. Fall, "A delay-tolerant network architecture for challenged internets," in *SIGCOMM '03: Proceedings of the 2003 conference*

on Applications, technologies, architectures, and protocols for computer communications, pp. 27–34, Karlsruhe, Germany, 2003.

[26] P. N. Pathirana, N. Bulusu, A. V. Savkin, and S. Jha, "Node localization using mobile robots in delay-tolerant sensor networks," *IEEE Transactions on Mobile Computing*, vol. 4, no. 3, pp. 285–296, 2005.

[27] A. Baggio and K. Langendoen, "Monte Carlo localization for mobile wireless sensor networks," *Ad Hoc Networks*, vol. 6, no. 5, pp. 718–733, 2008.

[28] C. Bettstetter, G. Resta, and P. Santi, "The node distribution of the random waypoint mobility model for wireless ad hoc networks," *IEEE Transactions on Mobile Computing*, vol. 2, no. 3, pp. 257–269, 2003.

[29] L. Hu and D. Evans, "Localization for mobile sensor networks," in *MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking*, pp. 45–57, Philadelphia, Pennsylvania, USA, 2004.

[30] Y. Liu, X. Zhang, X. Guan, and D. Delahaye, "Potential odor intensity grid based UAV path planning algorithm with particle swarm optimization approach," *Mathematical Problems in Engineering*, vol. 2016, Article ID 7802798, 16 pages, 2016.

[31] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, pp. 1942–1948, Perth, WA, Australia, 1995.