Xue *EURASIP Journal on Wireless Communications and Networking* (2019) 2019:221
https://doi.org/10.1186/s13638-019-1540-z

EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

Research on range-free location algorithm for wireless sensor network based on particle swarm optimization

Dalong Xue

Abstract

Location technology is the key support technology of wireless sensor network (WSN). The hop number and hop distance information obtained by traditional distance vector hop (DV-Hop) location algorithm can only be acquired by solving the nonlinear equations, and the solution of the equation determines the accuracy of node location. Although the least squares method has better estimation performance, the solution results are sensitive to the average hop distance, which will lead to the large error in the solution of the equation. In order to solve the problem of location error caused by initial value sensitivity of least squares method in the coordinate calculation stage of unknown nodes and beacon nodes, a range-free location algorithm based on particle swarm optimization (PSO) is proposed in this paper. The proposed approach solves the problem of location error caused by initial value sensitivity of least squares method, obtains relatively accurate solution, and improves the accuracy of location algorithm. The experimental results show that the PSO algorithm has faster convergence speed and higher location accuracy than the non-optimization algorithm.

Keywords: Wireless sensor network, Particle swarm optimization, DV-Hop algorithm, Least squares method

1 Introduction

Node location is a key technology in wireless sensor network, and its location accuracy is directly related to the overall performance of wireless sensor network (WSN) system [1, 2]. At present, the researches of WSN location algorithm have achieved rich results, and many novel solutions and ideas are used to solve the localization problem [3, 4]. Among them, distance vector hop (DV-Hop) location algorithm is the most widely used method. This method has the advantages of low complexity and good scalability, but its location accuracy is lower [5]. In the traditional DV-Hop algorithm, the minimum hops between beacon nodes and unknown nodes and the average hop distance between beacon nodes are the main reasons for location errors. At the same time, the hop number and hop distance information obtained in the calculation process of location algorithm usually use the least square method to solve the nonlinear equations in order to obtain the coordinates of unknown nodes. Although the least squares method has better estimation performance, the final solution of the equation will have a large deviation due to its sensitivity to the initial value [6]. To solve the above problems, many scholars optimized the nonlinear equations by using some optimization algorithms, such as annealing algorithm, ant colony algorithm, and so on [7]. Through the iterative correction ability of these optimization algorithms, the error caused by the sensitivity to initial values is reduced.

In recent years, particle swarm optimization (PSO), as a bio-evolutionary algorithm, has attracted much attention and played an important role in solving optimization problems. Compared with other genetic algorithms, PSO algorithm has no crossover and mutation operations, and it has fast search speed, high precision, good memory, and easy to be implemented in engineering. Many scholars have introduced PSO algorithm into DV-Hop algorithm. Because of its low sensitivity to measurement errors, PSO can achieve rapid optimization, thus the node location error is reduced [8]. In ref. [9], PSO is used to improve DV-Hop algorithm.

Correspondence: xue_dalong@outlook.com
School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China

The simulation results show that the location accuracy and location coverage are better than traditional DV-Hop location algorithm, but PSO algorithm is easy to fall into local optimum. In ref. [10], PSO is firstly introduced into the localization problem of wireless sensor network based on range-free method. Compared with the existing centroid localization method, it greatly improves the localization accuracy. However, when the nodes are distributed dispersedly in the network, the algorithm will cause large location errors. Reference [11] can use PSO to find the optimal location of the target node, but it increases the complexity of calculation, and affects the location effect of the node to some extent. Reference [12] proposes to optimize DV-Hop localization algorithm by using the connectivity difference between nodes. Moreover, the experimental results show that the algorithm has higher localization accuracy than traditional DV-Hop algorithm.

In order to solve the problem of location error caused by initial value sensitivity of least squares method in the coordinate calculation stage of unknown nodes and beacon nodes, a range-free location algorithm based on PSO is proposed in this paper. The proposed approach can solve the problem of location error caused by initial value sensitivity of least squares method, obtain relatively accurate solution, and improve the accuracy of location algorithm.

2 Description on location problem

Figure 1 shows the distribution of unknown nodes and beacon nodes. When the unknown node obtains more than three distances from the beacon nodes, the positioning accuracy can be improved by using the redundant distance information. At this time, the least square method can be used to estimate the position of the unknown node.

The location process of DV-Hop algorithm can be divided into three stages. In the first and second stages, the estimated distances from unknown node $O(x, y)$ to beacon nodes $A_1(x_1, y_1)$,$A_2(x_2, y_2)$,$A_3(x_3, y_3)$, ...,$A_n(x_n, y_n)$ are d_1, d_2, d_3,...,d_n respectively, and the ranging errors are ε_1, ε_2, ε_3, ..., ε_n respectively, which

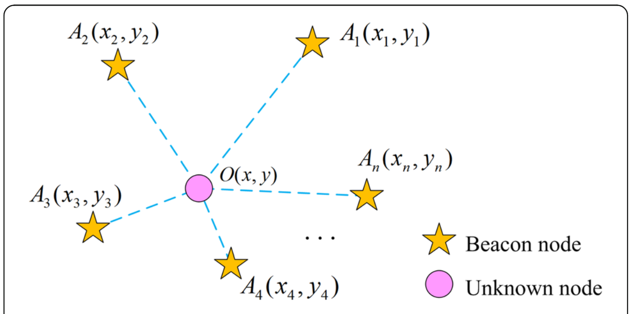


Fig. 1 Distribution diagram of unknown nodes and beacon nodes

can satisfy $\left|\sqrt{(x-x_i)^2+(y-y_i)^2}-d_i\right| \leq \varepsilon_i$, i = 1,2,3,...,n. After expansion, the following equations can be obtained:

$$\begin{cases} d_1-\varepsilon_1 \leq \sqrt{(x-x_1)^2+(y-y_1)^2} \leq d_1+\varepsilon_1 \\ d_2-\varepsilon_2 \leq \sqrt{(x-x_2)^2+(y-y_2)^2} \leq d_2+\varepsilon_2 \\ \quad\quad \mathrm{M} \\ d_n-\varepsilon_n \leq \sqrt{(x-x_n)^2+(y-y_n)^2} \leq d_n+\varepsilon_n \end{cases} \tag{1}$$

The smaller the sum of errors ε_1, ε_2, ε_3, ..., ε_n is, the more accurate the estimation position is, and the smaller the value of $f(x, y)$ in Eq. (2) is. Therefore, the location problem can be transformed into solving the minimum value problem of nonlinear equations, that is, solving the estimated coordinate (x, y) to minimize the value of $f(x, y)$ in Eq. (2). Fitness function is used to evaluate the quality of particle position and guide the search direction of the algorithm. Fitness function is defined as:

$$f(x, y)=\frac{1}{N}\sum_{i=1}^{N}\left|\sqrt{(x-x_i)^2+(y-y_i)^2}-d_i\right| \tag{2}$$

Where $f(x, y)$ is the fitness value of particle position (x, y), (x_i, y_i) is the position coordinate of beacon node i, and d_i is the distance from unknown node to beacon node i.

3 The classical PSO algorithm

3.1 The basic principle of PSO algorithm

In the optimization process of PSO algorithm, every possible optimal position is usually treated as a particle, and each particle has a related fitness function to control the motion benchmark of particle in the optimization process of algorithm. At the same time, in the process of particle moving to the optimal "food," there is also a vector velocity to control the moving direction and speed of the particle in optimization region. Each particle can update its position in the optimal solution space by iteration optimization model, and the updating method of particle position is shown in Fig. 2.

In Fig. 2, x is the initial position of the particle, v is the "flying" velocity of the particle, and p is the optimal position of the particle to be searched. In the process of searching for the optimal particle position iteratively, each particle updates and moves its position by recording and updating two key values which are closely related to its position in the current optimization space. The two key values are the best individual position and the best global position of particle.

Particle swarm is usually abstracted as a geometric model, i.e., assuming that there are N particles in a particle swarm, and the optimization space of the particle

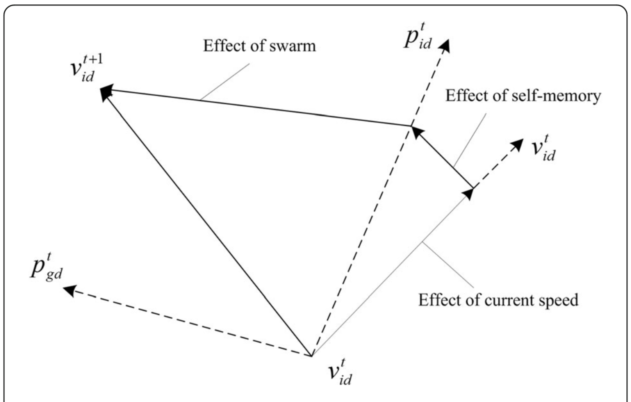


Fig. 2 Updating method of particle position

swarm is a *D*-dimensional space, then each particle in the particle swarm can be represented as the position vector, which is shown in Eq. (3):

$$X_i = (x_{i1}, x_{i2}, \Lambda, x_{iD}), \quad i = 1, 2, \Lambda, N \tag{3}$$

The velocity vector of each particle can be recorded as:

$$V_i = (v_{i1}, v_{i2}, \Lambda, x_{iD}), \quad i = 1, 2, \Lambda, N \tag{4}$$

The best individual location that each particle has experienced can be recorded as:

$$P_{best(i)} = (p_{i1}, p_{i2}, \Lambda, p_{iD}), \quad i = 1, 2, \Lambda, N \tag{5}$$

From the beginning iteration to the current iteration of particle, the best position for all particles can be recorded as follows:

$$g_{best} = \left(p_{g1}, p_{g2}, \Lambda, p_{gD}\right) \tag{6}$$

After obtaining the best individual position and the best global position of each particle, the velocity and position of each particle in the iterative optimization process can be updated according to the following two equations:

$$v_{id}^{t+1} = \omega \cdot v_{id}^{t} + c_1 r_1 \left(p_{id}^{t} - x_{id}^{t}\right) + c_2 r_2 \left(p_{gd}^{t} - x_{id}^{t}\right) \tag{7}$$

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t} \tag{8}$$

Where ω is the inertia weight of the particles, c_1 and c_2 are learning factors, also known as acceleration constants, and r_1 and r_2 are uniform random numbers in the range of [0,1].

3.2 Analysis on algorithm parameters

According to Eqs. (7) and (8), there are four key parameters in PSO algorithm: inertia weight ω, learning factors c_1 and c_2, which can control the velocity change during iteration process of particle, and then control the trajectory of particle in the whole optimization process. The maximum velocity $v_{\max}$ represents the position moving step of the particle in unit time, and it reflects the speed of searching for the optimal solution of the particle. Therefore, they have great influence on the whole PSO algorithm.

1. Inertia weight ω

The inertia weight coefficient weights the last iteration speed of the current particle, which can increase the searching ability of the particle toward the optimal solution in the current iteration process. Increasing the value of inertia weight can keep the particle moving toward the far potential optimal position. Meanwhile, reducing the value of inertia weight can weaken the searching ability of the particle in the optimization space, and it can only search the optimal value locally in the current region near the particle. When the inertia weight value is zero, the velocity updating of the particle in the current iteration process only depends on the best position that the individual has experienced and the global best position that all the particles have experienced, and it does not need to refer to the velocity value of the last particle, which results in the deviation of the particle in the process of searching for the optimal solution. Therefore, when the actual algorithm is solved iteratively, the value of the inertia weight needs to be adjusted, so as to control the particles to approach the optimal solution quickly, and to balance the iteration efficiency and optimization accuracy of the algorithm.

2. Learning factors c_1 and c_2

The learning factors c_1 and c_2 can control the effect of the best position experienced by each individual and the best position experienced by all particles on the velocity updating. Moreover, the particles can control the optimal path by referring to these two values. Without the existence of c_1 and c_2, the particle will fly straight at the initial speed until it reaches the boundary area of the optimal solution space, and the possibility of searching the optimal value is greatly reduced. In order to get the best solution of PSO in the process of optimization, the learning factors c_1 and c_2 should also change with the iteration condition of particle swarm.

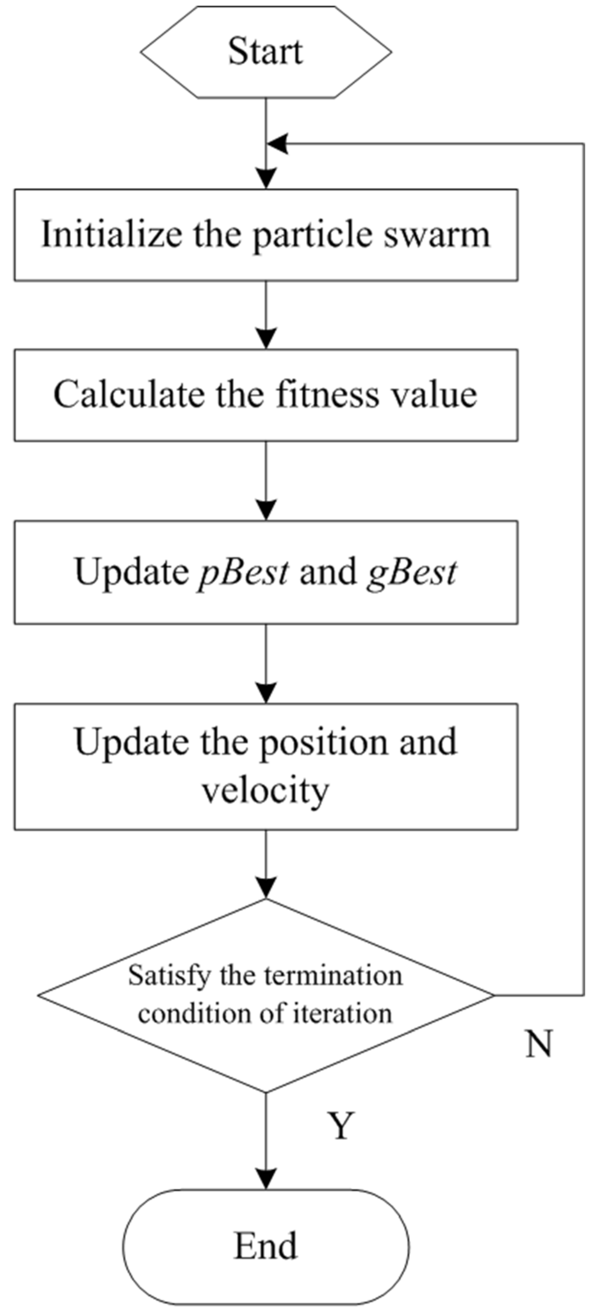
3. Maximum velocity of particle $v_{\max}$

During the iteration process, the maximum velocity $v_{\max}$ represents the position moving step of the particle in unit time, and it reflects the speed of searching for the optimal solution of the particle. In the iteration process of the actual algorithm, the maximum velocity of the particle in each dimension is always set as $v_{d\max} = k \times x_{d\max}$, $0.1 \le k \le 1$, where $x_{d\max}$ represents the length of each dimension of the optimization space.

3.3 The process of PSO algorithm

The concrete process of particle swarm optimization is shown in Fig. 3.

The specific steps of particle swarm optimization are as follows:



Start
Initialize the particle swarm
Calculate the fitness value
Update pBest and gBest
Update the position and velocity
Satisfy the termination condition of iteration
N
Y
End

Fig. 3 Flow chart of particle swarm optimization

Step 1: Assuming that the number of particles in the population is n, and these n particles are generated randomly. The velocity v_i and position x_i of the particles are assigned the initial values, and the individual optimal solution is the current position of the particles.

Step 2: According to Eq. (9), the fitness function values of all particles are obtained.

$$F(x,y) = \sqrt{(x-x_i)^2 + (y-y_i)^2} - d_i \tag{9}$$

Step 3: According to Eqs. (10) and (11), the individual and global optimal values of particles are updated.

$$pBest^{t+1} = \begin{cases} x^{t+1}, & if \quad F\left(X^{t+1}\right) \le F\left(pBest^t\right) \\ pBest^{t+1}, & if \quad F\left(X^{t+1}\right) > F\left(pBest^t\right) \end{cases} \tag{10}$$

$$gBest^{t+1} = \begin{cases} pBest^{t+1}, & if \quad F\left(pBest^{t+1}\right) \le F\left(gBest^t\right) \\ gBest^t, & if \quad F\left(pBest^{t+1}\right) > F\left(gBest^t\right) \end{cases} \tag{11}$$

Step 4: The velocity and position of particles are updated according to Eqs. (7) and (8).

Step 5: Finding the fitness function that meets the conditions or the number of iterations will terminate the algorithm. Otherwise, continue step 2 to find the optimal particle.

Step 6: The particle corresponding to the global optimal value is obtained as the estimated position of the unknown node.

4 The improved PSO algorithm

4.1 The improvement on algorithm

In the standard POS algorithm, the learning factors c1 and c2 are generally set as two fixed parameters, which have a great impact on the algorithm. In different periods of population evolution, the search performance of particles is different. Generally speaking, in the global search scenario, the algorithm should have better search performance for the initial stage; in the end stage of the algorithm, the algorithm should have better development ability to improve the convergence speed of the algorithm. If a fixed acceleration coefficient is used in the algorithm, it will inevitably limit the ability of particles to adjust the search step and flight direction in flight period. Therefore, in this paper, two acceleration factors are added to the traditional POS algorithm, which can adapt to change.

$$A_{i1} = \ln\left(\left|\frac{f(P_i(t)) - f(X_i(t))}{\overline{PF}}\right| + e - 1\right) \tag{12}$$

$$A_{i2} = \ln\left(\left|\frac{f(P_g(t))-f(X_i(t))}{\overline{GF}}\right| + e-1\right) \tag{13}$$

Where $\overline{PF} = \frac{1}{n}\sum_{i=1}^{n}(f(P_i(t))-f(X_i(t)))$ and $\overline{GF} = \frac{1}{n}\sum_{i=1}^{n}(f(P_g(t))-f(X_i(t)))$. Moreover, n is the number of population, $f(x)$ is the objective function, e is the base of natural logarithm, and $\overline{PF}$ represents the mean of the difference between the individual fitness and the optimal value of a single particle in a population. Therefore, if $|f(P_i(t))-f(X_i(t)) > |\overline{PF}||$ exists, there will be $A_{i1} > 1$, which means the particle will be accelerated. Meanwhile, if $|f(P_i(t))-f(X_i(t)) < |\overline{PF}||$ exists, there will be $A_{i1} < 1$, which means the particle will be decelerated. Similarly, $\overline{GF}$ represents the mean value of the difference between individual fitness and population optimal values of all particles in a population. Therefore, if $|f(P_g(t))-f(X_i(t)) > |\overline{GF}||$ exists, there will be $A_{i2} > 1$, which means the particle will be accelerated. Meanwhile, if $|f(P_g(t))-f(X_i(t)) < |\overline{GF}||$ exists, there will be $A_{i2} < 1$, which means the particle will be decelerated.

The cognitive and social coefficients used in this paper are shown as follows:

$$\omega_1 = \frac{c_1^2}{c_1 + c_2} \tag{14}$$

$$\omega_2 = \frac{c_2^2}{c_1 + c_2} \tag{15}$$

Where c_1 and c_2 are the same as the acceleration constants of traditional PSO algorithm.

In the traditional PSO algorithm, $\omega(t)$ decreases linearly along with the increase of the number of evolutions.

$$\omega(t) = \frac{(\omega_i-\omega_c)(T_{\max}-t)}{T_{\max}} + \omega_c \tag{16}$$

Where t is the current iteration number, ω_i is the initial inertia weight, $T_{\max}$ is the largest iteration number, ω_c is the inertia weight when iterating to the algebraic maximum, and $\omega_{\max}$ = 0.9 and $\omega_{\min}$ = 0.4. With the increase of iteration, $\omega(t)$ will gradually decrease, which makes the algorithm have stronger global search performance in the initial stage and increases the possibility of obtaining global optimal solution. At the end of the algorithm, the smaller weight can give the particle stronger local search performance and improve the convergence speed of the algorithm.

The influence of inertia weight ω on PSO algorithm is shown as follows: when the value of ω is too large, it can prevent the algorithm from entering the local optimal result. Meanwhile, when the value of ω is too small, it can make the algorithm search locally more effectively and improve the convergence speed of the algorithm.

According to the above analysis, this paper proposes that the inertia weight is a random, linear, and decreasing weight in the later iteration stage, so that the ω of particle will get a larger value in the initial stage of the algorithm in order to guarantee the diversity of the particle. Meanwhile, at the end of the algorithm, the ω of particle are likely to get larger, so as to get out of the local extremum. The value of random inertia weight ω varies with the number of iterations:

$$\omega = \omega_{\max} - \frac{(\omega_{\max}-\omega_{\min}) \cdot t}{T_{\max}} \tag{17}$$

When the number of iterations is larger than $0.7T_{\max}$, there is $\omega = 0.4 + 0.3 * rand()$. The function $rand(\ )$ is used to generate a random number that is evenly distributed between 0 and 1.

Then the equation of inertia weight ω for the whole process of the algorithm is shown as follows:

$$\omega(t) = \begin{cases} \omega_{\max} - \frac{(\omega_{\max}-\omega_{\min}) \cdot t}{T_{\max}}, t < 0.7T_{\max} \\ 0.4 + 0.3 * \ rand(), else \end{cases} \tag{18}$$

Therefore, the evolution equation of the adaptive PSO algorithm is shown as follows:

$$\begin{aligned} v_{id}^{k+1} &= \omega(t) \cdot v_{id}^{k} + \omega_1 \cdot \ rand()A_{i1} \cdot \left(p_{id}-x_{id}^{k}\right) \\ &\quad + \omega_2 \cdot \ rand()A_{i2} \cdot \left(p_{gbest}-x_{id}^{k}\right) \end{aligned} \tag{19}$$

$$x_{id}^{k+1} = x_{id}^{k} + v_{id}^{k+1} \tag{20}$$

4.2 Selection of fitness function

When estimating the distance between unknown node and beacon nodes in DV-Hop algorithm, the average hop distance and the hops of beacon nodes between two nodes are mainly used to calculate it, which has certain errors. Assuming that the beacon nodes are $P_1(x_1, y_1)$, $P_1(x_2, y_2)$, ..., $P_1(x_n, y_n)$, the estimated distances between unknown node and each beacon node obtained by step 1 and step 2 of DV-Hop algorithm are respectively d_1, d_2, d_3,...,d_n, and the differences between estimated distance and real distance are respectively ε_1, ε_2, ε_3, ..., ε_n. From the previous introduction, we can get the equations shown in Eq. (21):

$$\begin{cases} \sqrt{(\hat{x}-x_1)^2+(\hat{y}-y_1)^2}=d_1+\varepsilon_1 \\ \sqrt{(\hat{x}-x_2)^2+(\hat{y}-y_2)^2}=d_2+\varepsilon_2 \\ \quad \text{M} \\ \sqrt{(\hat{x}-x_n)^2+(\hat{y}-y_n)^2}=d_n+\varepsilon_n \end{cases} \tag{21}$$

The coordinate $(\hat{x}, \hat{y})$ of unknown node should satisfy all the equations mentioned above. When the sum of $|\varepsilon_1|$,$|\varepsilon_2|$,..., and $|\varepsilon_n|$ is smaller, the accuracy of estimating coordinate $(\hat{x}, \hat{y})$ is higher. Therefore, to some extent, the optimization of PSO algorithm for estimating coordinate of unknown node can be transformed into the process of minimizing Eq. (22).

$$f_i(\hat{x},\hat{y}) = \left|\sqrt{(\hat{x}-x_i)^2+(\hat{y}-y_i)^2}-d_i\right| \tag{22}$$

PSO algorithm uses the current particle fitness value to distinguish the advantages and disadvantages of its specific location in the optimization stage. In the current population, any particle is the coordinate possible solution of the unknown node. This paper mainly distinguishes the advantages and disadvantages of any particle through the current fitness function expressed by Eq. (23):

$$\text{fitness}(\hat{x},\hat{y}) = \sum_{i=1}^{n}\left(\frac{f_i(\hat{x},\hat{y})}{h_i}\right)^2 \tag{23}$$

Where n denotes the number of beacon nodes, and h_i denotes the number of hops from unknown nodes to beacon nodes. In order to avoid excessive hops between unknown nodes and beacon nodes, which results in affecting estimation distance too much from cumulative errors, h_i is added into the fitness function in order to reduce the impact of excessive hops. In contrast, when h_i becomes larger, the influence of the beacon node on the accuracy of fitness becomes smaller.

The localization process of the improved algorithm is shown in Fig. 4.

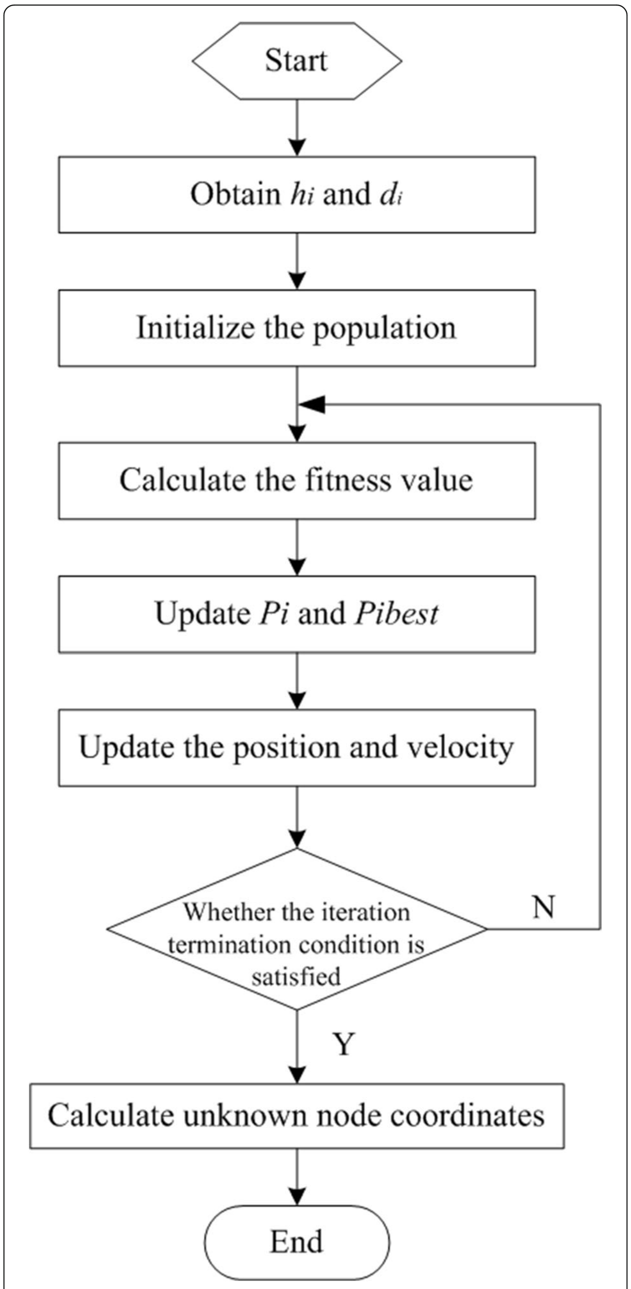

Start
Obtain hi and di
Initialize the population
Calculate the fitness value
Update Pi and Pibest
Update the position and velocity
Whether the iteration termination condition is satisfied
N
Y
Calculate unknown node coordinates
End

Fig. 4 Flow chart of the DV-Hop algorithm based on improved PSO

5 Experimental results and analysis

In order to verify the location effect of PSO optimization algorithm, the traditional DV-Hop location algorithm and the improved PSO-based DV-Hop algorithm are simulated on the Matlab experimental platform. In the same network environment, the performance of the algorithm is compared by changing beacon nodes, communication radius.

The simulation environment is shown as follows: (1) all sensor nodes are arbitrarily arranged in the network environment; (2) the number of unknown nodes is 200, and the communication radius of nodes is 15 m; (3) the number of beacon nodes is 30, and they are arbitrarily distributed in the monitoring area, and each broadcast radiation distance can reach the whole network. In the network, the population size is 20 and the maximum number of evolutionary iterations of particles in the particle swarm is 300. According to Eq. (18), the inertia weight ω is changed from $\omega_{\max}$ = 0.9 to $\omega_{\min}$ = 0.4, and the learning factors are chosen as c_1 = c_2 = 2. Figure 5 shows the scatter graph of nodes.

5.1 Influences of beacon node density on algorithm

Each group of data is the result after 60 times average. In the network topology shown in Fig. 4, the communication

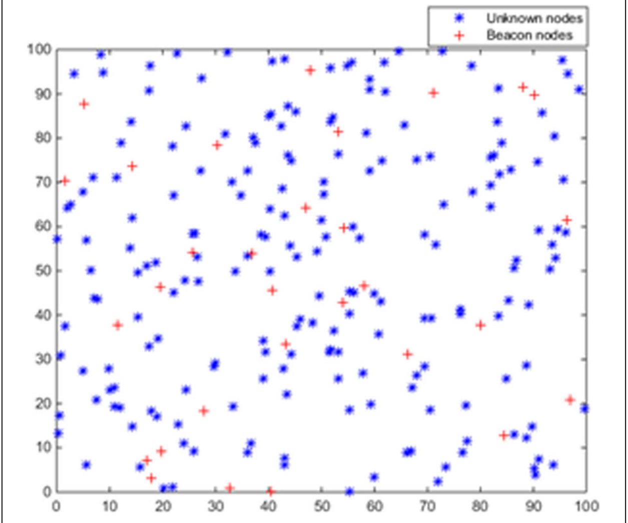


Fig. 5 The scatter graph of nodes

radius R = 15 m is simulated when proportion of beacon nodes from 5 to 30% of the total number of nodes. The experimental results are shown in Fig. 6.

Under the condition of the same beacon node density, the error rate of the improved PSO DV-Hop algorithm is not greater than that of the traditional DV-Hop algorithm. When the proportion of beacon nodes is 10–25%, the momentum of improving the location accuracy is obvious for PSO optimization algorithm; when the density of beacon nodes exceeds 25%, the location accuracy of the algorithm increases slowly. Moreover, the increasing trend of traditional DV-Hop algorithm is also slowing down, which shows that when the number of beacon nodes in the environment is large, the location accuracy of the algorithm tends to be stable. When the density of beacon nodes is the same, the location accuracy of the improved algorithm is better than that of the traditional DV-Hop algorithm in the network environment.

5.2 Influences of communication radius on algorithm

In the same experiment simulation, we mainly depend on changing the communication radius of unknown nodes to investigate the influence of network connectivity on the algorithm. The total number of nodes scattered in the environment is 200, the sparse density of beacon nodes is adjusted to 20%, and the transformation range of communication radius is [15 m, 40 m]. The simulation results are shown in Fig. 7.

Figure 7 shows that the improved PSO algorithm outperforms the traditional DV-Hop algorithm in location accuracy when the communication radius is small. In the process of changing the communication radius from 15 to 40 m, the location accuracy of the two algorithms shows an improving trend, and the attenuation degree of the location error of the improved algorithm is still better than that of the traditional DV-Hop algorithm.

6 Conclusion

In order to solve the problem of location error caused by initial value sensitivity of least squares method in the coordinate calculation stage of unknown nodes and beacon nodes, a range-free location algorithm based on particle swarm optimization (PSO) is proposed in this paper. The proposed approach solves the problem of location error caused by initial value sensitivity of least squares method, obtains relatively accurate solution, and improves the accuracy of location algorithm. In order to verify the location effect of PSO optimization algorithm,

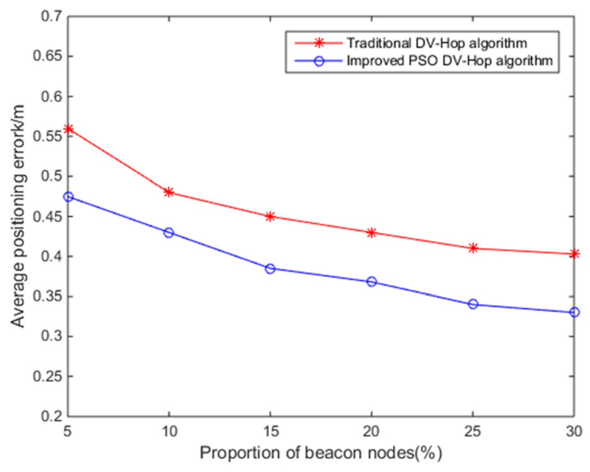


Fig. 6 Average location errors with different beacon node densities

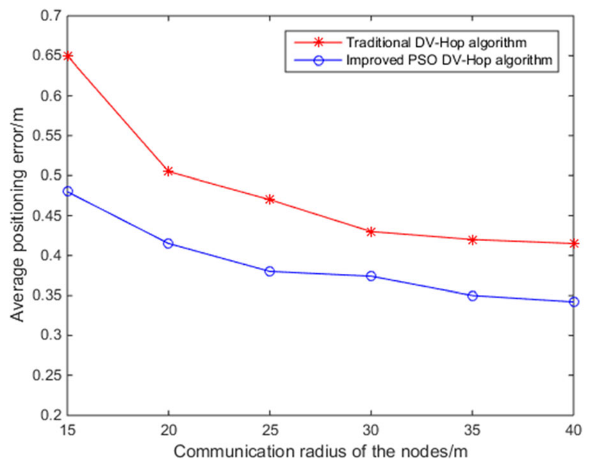


Fig. 7 Average location errors with different communication radius

the traditional DV-Hop location algorithm and the improved PSO-based DV-Hop algorithm are simulated on the Matlab experimental platform. In the same network environment, the performance of the algorithm is compared by changing beacon nodes, communication radius.

The experimental results show that the PSO algorithm has faster convergence speed and higher location accuracy than the non-optimization algorithm.

Abbreviation

WSN: Wireless sensor network; DV-Hop: Distance vector hop; PSO: Particle swarm optimization

Acknowledgements

Not applicable.

Authors' contributions

The author completed the experiment and manuscript. The author read and approved the final manuscript.

Authors' information

Dalong Xue: PhD student, School of Computer Science and Technology, Beijing Institute of Technology. His research interests include computer network technology, software algorithm.

Funding

Not applicable.

Availability of data and materials

The data generated and analyzed during this study are included in this published article, and its supplementary information is also available from the corresponding author on reasonable request.

Competing interests

The author declares that he has no competing interest.

Received: 19 June 2019 Accepted: 16 August 2019
Published online: 03 September 2019

References

1. F.L. Lewis, Wireless sensor networks[J]. Smart Environ. Technol. Protoc. Appl. **181**(1), 11–46 (2016)
2. R. Shahbazian, S.A. Ghorashi, Distributed cooperative target detection and localization in decentralized wireless sensor networks[J]. J. Supercomput. **73**(4), 1715–1732 (2017)
3. M. Zhang, D. Zhang, F. Goerlandt, X. Yan, P. Kujala, Use of HFACS and fault tree model for collision risk factors analysis of icebreaker assistance in ice-covered waters. Saf. Sci. **111**, 128–143 (2019)
4. J. Fernandez-Bes, J. Arenas-Garca, M.T.M. Silva, et al., L. A. Azpicueta-Ruiz, Adaptive diffusion schemes for heterogeneous networks[J]. IEEE Trans. Signal Process. **65**(21), 5661–5674 (2017)
5. N.A.M. Maung, M. Kawai, Experimental evaluations of RSS threshold-based optimised DV-HOP localisation for wireless ad-hoc networks [J]. Electron. Lett. **50**(17), 1246–1248 (2014)
6. G. Song, D. Tam, Two novel DV-Hop localization algorithms for randomly deployed wireless sensor networks[J]. Int. J. Distrib. Sens. Netw. **9**, 1–12 (2015)
7. M.R. Tanweer, R. Auditya, S. Suresh, Directionally driven self-regulating particle swarm optimization algorithm[J]. Swarm Evol. Comput. **28**, 98–116 (2016)
8. Y. Zhang, J. Liang, S. Jiang, et al., A localization method for underwater wireless sensor networks based on mobility prediction and particle swarm optimization algorithms[J]. Sensors **16**(2), 212.GB/T 7714 (2016)
9. Z. Fengrong, Positioning research for wireless sensor networks based on PSO algorithm[J]. Elektronika Ir Elektrotechnika **19**(9), 7–10 (2013)
10. H. Bao, B. Zhang, C. Li, et al., Mobile anchor assisted particle swarm optimization (PSO) based localization algorithms for wireless sensor networks[J]. Wirel. Commun. Mob. Comput. **12**(15), 1313–1325 (2012)
11. H.A. Nguyen, H. Guo, K.S. Low, Real-time estimation of sensor node's position using particle swarm optimization with log-barrier constraint[J]. Instrum. Meas. IEEE Trans. **60**(11), 3619–3628 (2011)
12. L. Gui, T. Val, A. Wei, et al., Improvement of range-free localization technology by a novel DV-hop protocol in wireless sensor networks[J]. Ad Hoc Netw. **24**, 55–73 (2015)

Publisher's Note