Fast triangle flip bat algorithm based on curve strategy and rank transformation to improve DV-Hop performance

Xingjuan Cai[1*], Shaojin Geng[1], Penghong Wang[1], Lei Wang[2] and Qidi Wu[2]
1 Taiyuan University of Science and Technology, Complex System and Computation Intelligent Laboratory
Taiyuan, Shanxi 030024 - China.
[e-mail: xingjuancai@163.com, shaojin_geng@163.com, penghongwang@sina.cn]
2 Tongji University, Department of Control Science and Engineering
Shanghai 201804 - China.
[e-mail: wanglei@tongji.edu.cn, icslab2@tongji.edu.cn]
*Corresponding author: Xingjuan Cai

Received April 14, 2019; revised June 5, 2019; accepted June 23, 2019;
published December 31, 2019

Abstract

The information of localization is a fundamental requirement in wireless sensor network (WSN). The method of distance vector-hop (DV-Hop), a range-free localization algorithm, can locate the ordinary nodes by utilizing the connectivity and multi-hop transmission. However, the error of the estimated distance between the beacon nodes and ordinary nodes is too large. In order to enhance the positioning precision of DV-Hop, fast triangle flip bat algorithm, which is based on curve strategy and rank transformation (FTBA-TCR) is proposed. The rank is introduced to directly select individuals in the population of each generation, which arranges all individuals according to their merits and a threshold is set to get the better solution. To test the algorithm performance, the CEC2013 test suite is used to check out the algorithm's performance. Meanwhile, there are four other algorithms are compared with the proposed algorithm. The results show that our algorithm is greater than other algorithms. And this algorithm is used to enhance the performance of DV-Hop algorithm. The results show that the proposed algorithm receives the lower average localization error and the best performance by comparing with the other algorithms.

Keywords: DV-Hop, fast triangle flip, bat algorithm, threshold, rank transformation

1. Introduction

Wireless sensor network (WSN) [1] is a system, which mainly compose of three parts: node, gateway and software. It is an important issue [2] for WSN [3] to locate the information of nodes accurately [4]. Such as mine search and rescue [5], the coordinates of the position should be obtained when survivors are found. In view of the cost, most wireless nodes will not install GPS and other localization devices. Therefore, how to use fewer of the beacon nodes (nodes with GSP localization device) to predict the location of other ordinary nodes (nodes without localization device) has become an urgent problem [6] to be solved.

Due to the location information is crucial to WSN [7], various localization algorithms [8] have been proposed to improve the positioning accuracy of sensor nodes. And the classification of range-free and range-based algorithm is distinguished according to whether the distances need to calculate accurately. The range-based locating algorithm includes time of arrival algorithm (TOA) [9], Ad hoc positioning system (APS) [10] and received signal strength indicator algorithm (RSSI) [11], also the range-free locating algorithm includes the distance vector hop algorithm (DV-Hop) [12], convex position estimation (CPE) [13], and multi-dimensional scaling (MDS) [14]. The DV-Hop [15] positioning processes are implemented by multi-hop information and distance estimation. The distance between the ordinary node and beacon node is obtained by the following process. Firstly, the ordinary node records the minimum hops to the beacon node, and then calculate roughly the average distance per hop. Finally, the required distance is estimated according to the average distance per hop and the minimum number of hops. To get the accurate value, variety optimization algorithms are used to reduce the error of position [16], such as trilateral measuring.

Intelligent optimization algorithm [17] is a method constructed based on human cognition and learning experience of nature, which used to solve the complex optimization problem [18]. And in recent years, the bio-inspired optimization algorithm [19] has been applied to various fields [20]. Such as bat algorithm (BA) [21], particle swarm optimization (PSO) [22], firefly algorithm (FA) [23], cuckoo search (CS) [24] and pigeon-inspired optimization algorithm (PIO) [25]. Specifically, a new firefly inspired strategy [26] is proposed to spread and disseminate game in online social networks (OSNs) and decrease the acceptance-discontinuance anomaly. A many-objective optimization algorithm to protect the Privacy protection [27]. An improved cuckoo search algorithm to solve the problem of integer program [28]. A firefly algorithm is used to find the fixed point of a nonlinear function [29]. And most optimization algorithms [30] can be applied to practical problems [31], such as support vector machines [32], 0-1 knapsack problem [33] and IP assignment [34]. A new search algorithm based on cuckoo [35] is proposed to enhance DV-Hop performance. And a hybrid PSO with mutation (HPSOM) [36] is used to detect the code smell.

Bat algorithm [37], as a kind of swarm intelligence algorithm [38], simulates the echolocation prey behavior of bats to achieve search the optimal solution. And the mode of echolocation behavior is that each bat individual is regarded as a solution of the current the feasible region, each solution can be regarded as a fitness value. Each bat individual chooses a global or local search method according to probability. That is, when the given probability is satisfied, the global search mode is adopted; otherwise, search by local optimization. And each bat follows the current optimal bats by adjusting three parameters, including the pulse wave length, volume, and pulse emissivity. In this way, we can get the optimal solution in searching space. In a word, both of the global and the local search mode adopt the method of random

transformation, which means the individuals of the local search in each generation adopt the method of random selection to determine, and the proportion of the local search is determined by the pulse transmission frequency.

Cai [39] proposed the fast bat algorithm which adopted the triangle reversal curve strategy (FTBA-TC), which is combined with the fast triangular flip and curve decline strategy. In this paper, we continue to improve FTBA-TC so that the algorithm can balance the proportion of global and local searches. The contribution of this paper is as follows: 1) The rank-based transformation strategy is designed which sort the individual by the fitness value. 2) A threshold is set to dynamically adjust the proportion of performing global search and local search throughout the iteration process, which means that the threshold would change with the increase of the iterator. 3) Both experiments of CEC2013 and DV-Hop are used to demonstrate the proposed algorithm has the best performance.

The other parts of this paper are introduced as follows: Section 2 describes the localization algorithm of dv-hop. Section 3 describes the idea and implementation process of standard BA in detail. In addition, the main idea of rank-based transformation strategy is introduced emphatically. Section 4 tests our algorithm and applies it to wireless sensor node location. It is effective in improving the accuracy of DV-Hop algorithm that the proposed FTBA-TCR. At the end of the article, the conclusion is drawn in section 5.

2. DV-Hop Localization Algorithm

With the quick development of wireless communication [40], it is crucial to WSN [41] that the position information is attracting more and more attention. The algorithm of range-free and range-based as the branched of the WSN are illustrated in **Fig. 1**.

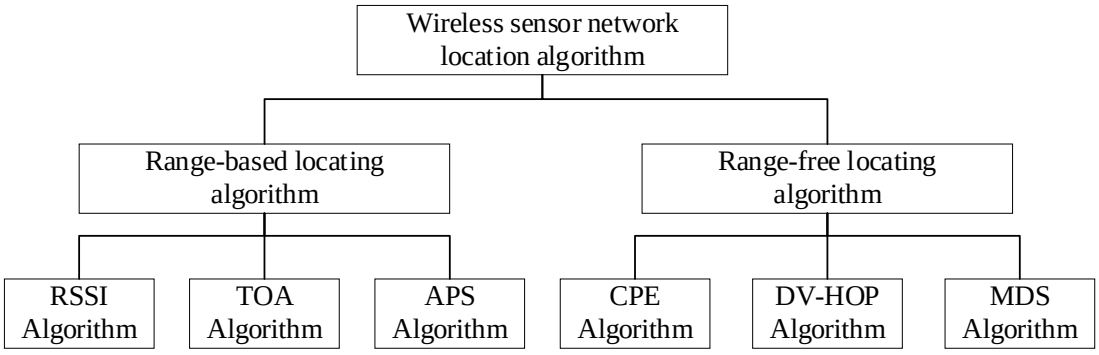

Fig. 1. Classification diagram of localization algorithm

The fundamentals of DV-Hop algorithm [42] are showed as follows: (1) the average distance per hop of beacon node is estimated according to the information of hop count and distance between beacon nodes; (2) the ordinary node is located by the estimation distance. Where $(B_1, B_2, B_3 \cdots, B_m)$ indicate m beacon nodes, $(U_1, U_2, U_3 \cdots, U_n)$ represent n ordinary nodes. The wireless sensor network has $m+n$ nodes, consider per hop average distance of the beacon node B_i, after permitting all beacon nodes to broadcast in the network, a certain number of hops (the number of nodes transmitted) are passed. The information of other nodes can be obtained by beacon node B_i. $(h_1, h_2, \cdots, h_{i-1}, h_{i+1}, \cdots, h_m)$ represent the number of hops

between beacon node B_i and B_m. Since the coordinates of beacon nodes is known, the average distance of per hop can be obtained as:

$$Hop_i = \frac{\sum_{k \neq i} \sqrt{(x_i - x_k)^2 + (y_i - y_k)^2}}{\sum_{k \neq i} h_k} \tag{1}$$

where (x_k, y_k) denotes the coordinates of beacon node B_k.

In the broadcast process, B_i may receive multiple hops of B_k with different paths and nodes. At this time, we only keep the minimum hops received.

Meanwhile, the relationship among of the beacon nodes, the ordinary nodes and the estimated distance can be expressed as follows:

$$\begin{cases} (x - x_1)^2 + (y - y_1)^2 = d_1^2 \\ (x - x_2)^2 + (y - y_2)^2 = d_2^2 \\ \qquad \cdots\cdots \\ (x - x_m)^2 + (y - y_m)^2 = d_m^2 \end{cases} \tag{2}$$

$$d_k = Hop_k \cdot T_k \tag{3}$$

where, d_k is the estimated distance, the following objective function is obtained by this distance:

$$f(x, y) = \sum_{k=1}^{m} \alpha_k \cdot \left| d_k^2 - (x - x_k)^2 - (y - y_k)^2 \right| \tag{4}$$

where, the value of α_k is the reciprocal of the number of hops, which means the larger the number of hops, the smaller the value of α_k.

3. Fast Triangle Flip Bat Algorithm Based on Curve Strategy and Rank

3.1 Standard Bat Algorithm

A heuristic intelligent algorithm [43], bat algorithm [44], simulates the principle of echolocation in bat predation and has the advantages of simple structure, few parameters, strong robustness, easy understanding and implementation. Therefore, it has received extensive attention and has become a hot spot in the field of computational intelligence research [45]. Bat algorithm [46] is designed by Yang in 2010.

For the minimum objective function min $f(x)$, the variable is $x = (x_1, x_2, \cdots, x_k)$. The attributes of each bats individual is defined as follows:

$$\{v_k(t), x_k(t), f_k(t), r_k(t), A_k(t)\} \tag{5}$$

where $v_k(t)$ and $x_k(t)$ denotes the velocity and position of the *kth* bat in *t* generation, respectively. Meanwhile $f_k(t)$, $r_k(t)$ and $A_k(t)$ denotes the frequency, the rate of pulse emission and the loudness, respectively.

. Firstly, the population is initialized, which means that the bat flies at position $x_k(t)$ with velocity $v_k(t)$. Meanwhile, the velocity and position are updated by adjusting the frequency. The updating equation is:

$$x_k(t) = x_k(t-1) + v_k(t) \tag{6}$$

$$v_k(t) = v_k(t-1) + (x_k(t) - x_{best}(t)) \cdot f_k(t) \tag{7}$$

$$f_k(t) = f_{\min} + (f_{\max} - f_{\min}) \cdot \sigma \tag{8}$$

where $x_{best}(t)$ represents the optimal position in t generation, and $\sigma \in [0,\ 1]$.

And the local search strategy is showed as follows:

$$x_k(t+1) = x_{best}(t) + \delta_k \cdot \bar{A}_k(t), \quad if \quad \eta > r_k(t) \tag{9}$$

where $\delta_k \in [-1,\ 1]$, $\eta \in [0,\ 1]$ and $\bar{A}_k(t)$ denotes the average loudness.

The new position is updated only when the conditions are met:

$$\boldsymbol{x}_k(t) = \begin{cases} \boldsymbol{x}'_k(t) & if\ \beta < A_k(t)\ and\ f(\boldsymbol{x}'_k(t)) < f(\boldsymbol{x}_k(t)) \\ \boldsymbol{x}_k(t-1) & otherwise \end{cases} \tag{10}$$

where $\beta \in [0,\ 1]$ and $\boldsymbol{x}'_k(t)$ denotes the new position updated by Eq. (6) and Eq. (9).

Furthermore, the updating equations of frequency, rate of pulse emission and loudness are showed as follows:

$$A_k(t) = \alpha A_k(t-1) \tag{11}$$

$$r_k(t+1) = r(0)[1 - \exp(-\gamma t)] \tag{12}$$

where α and γ are predefined parameters, $A(0)$ and $r(0)$ are two intial values of loudness and pulse emission, respectively.

The flow chart of the BA is showed in **Fig. 2**.

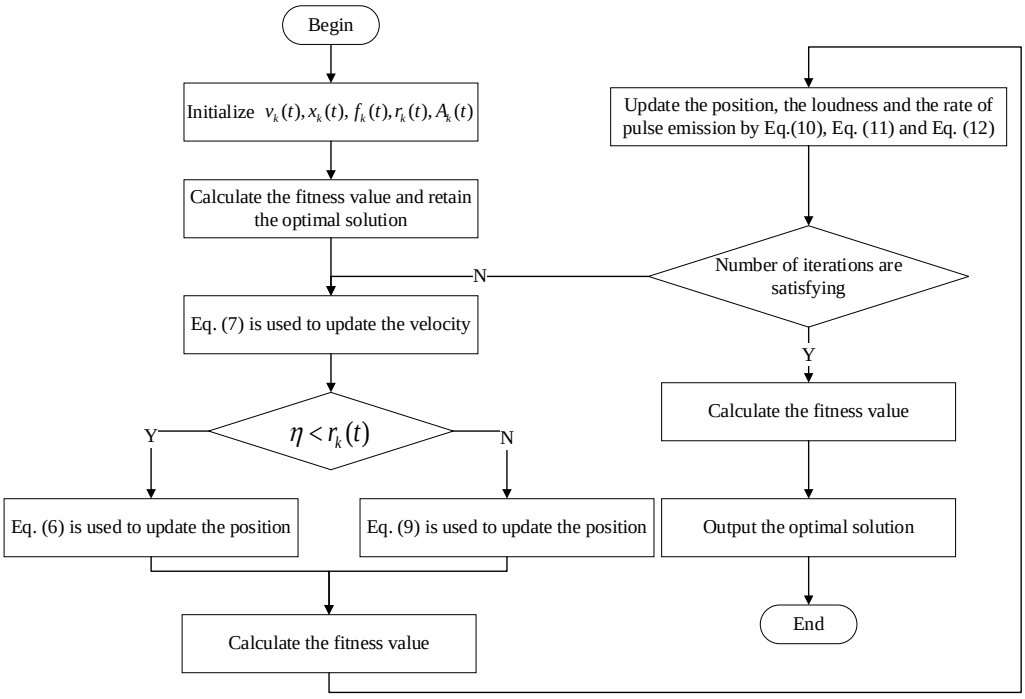

Fig. 2. The standard BA

3.2 Rank-based Transformation Strategy

The fast bat algorithm with triangle flip (FTBA) [47] improves global optimization ability validly. In addition, the FTBA-TC is combining the fast triangular flip and curve decline

strategy, which modify the local search ability of the algorithm. According to FTBA, the velocity update equation is showed as follows:

$$\boldsymbol{v}_k(t) = (\boldsymbol{x}_{\boldsymbol{m}}(t-1) - \boldsymbol{x}_{\boldsymbol{u}}(t-1)) \cdot f_k(t-1) \tag{13}$$

$$\boldsymbol{v}_k(t) = (x_{best}(t-1) - \boldsymbol{x}_{\boldsymbol{m}}(t-1)) \cdot f_k(t) \tag{14}$$

where $x_m(t-1)$ and $x_u(t-1)$ refer to two randomly positions in the current generation.

Based on FTBA-TC [39], the curve decline strategy uses a disturbance parameter $\tau \cdot x_{\max}$ to replace the average loudness $\overline{A}(t)$.

$$\boldsymbol{x}(t) = \boldsymbol{x}_{best}(t-1) + \boldsymbol{\varepsilon}_k \tau(t-1) \cdot x_{\max} \tag{15}$$

$$\tau(t) = \tau_{\max} \cdot [1 - (\frac{\tau_{\max} - \tau_{\min}}{\tau_{\max} \cdot (LG-1)} \cdot (t-1))^{k1}]^{k2} \tag{16}$$

where ε_k is a random vector within $[-1,\ 1]$ that satisfies uniform distribution, $\tau(t-1) \cdot x_{\max}$ is the area search radius, $\tau(t-1)$ decreases linearly with the number of evolutionary iterations increases, t denotes the number of evolutionary iterations, LG represents the maximum number of evolutionary iterations, the value of $k1$ and $k2$ determined the down trend. The algorithm has the best performance when $k1 = 1$ and $k2 = 4$ through this experiments.

In addition, in each generation of bat algorithm, all individuals are arranged with descending order according to the merits and demerits of the adaptive values to obtain the ranking of each bat, which is called the rank of the bat, and the rank is a commonly used statistic in statistics. Generally speaking, the global optimal position is less likely to be in the vicinity of individuals with poor fitness values. Therefore, some individuals with poor global optimal position can adopt the local perturbation mode to carry out mining operations. The $Threshold$ is designed for each generation, if the bats ranked below $Threshold \times Popsize$ (the $Popsize$ represents the number of bats in a population), local search was used, otherwise, global search was used.

In the early stage of the algorithm, a large area of global optimization is required to determine the optimal position and the optimal position will be exploitated in the later stage. Therefore, in the early stage, a large number of bat individuals are required to conduct global optimization, while in the later stage, a large number of individuals are required to conduct local optimization through disturbance. Therefore, the value of $Threshold$ will increase with the increase of the iteration.

$$Threshold = Th_{\min} + (Th_{\max} - Th_{\min}) \times \frac{t-1}{LG-1} \tag{17}$$

where, $Th_{\min}$ is the lower bound of $Threshold$, $Th_{\max}$ is the upper bound of $Threshold$, t is the current generation, LG is the maximum evolutionary generations. Eq. (17) indicates that $Threshold$ will increase linearly from $Th_{\min}$ to $Th_{\max}$ with the increase of the evolutionary generations.

The transformation strategy is adopted in FTBA-TCR, which is showed in Eq. (17). And the pseudo-code of the FTBA-TCR is showed as (FTBA-TCR).

FTBA-TCR

```
Begin
  Initialize v_k(t), x_k(t), f_k(t), r_k(t) and A_k(t)
  Calculate the fitness value of each bat and select the best solution
  While (t < LG)
    If t < 0.258 · LG
      Eq. (13) is used to update the velocity for each bat
    Else
      Eq. (14) is used to update the velocity for each bat
    End
    Sort the fitness values
    If the individual's fitness value is weakness, the formula is expressed
    as Rank(f(x_k(t))) < Threshold × Popsize
      The local search strategy in Eq. (15) is used to update the position
    Else
      Eq. (6) is used to update the position
    End
    If f(x_k(t+1)) < f(x_k(t))
      Update the bat individual position with x_k(t+1) = x_k(t)
    End
    Update and save the position of the best solution
  End
  Output the best solution
End
```

4. Experiments

4.1 FTBA-TCR

In order to verify the performance of these strategies, these algorithms are tested in the CEC2013 test suite. The upper limit of parameter *Threshold* was $Th_{\max}$, and the lower limit of parameter *Threshold* was $Th_{\min}$. Since *Threshold* is a value within $[0,\ 1]$, the upper limit $Th_{\max}$ was tested at 0.5, 0.6, 0.7, 0.8, 0.9, while the lower limit $Th_{\min}$ was tested at 0.1, 0.2, 0.3, and 0.4. Therefore, there are 20 combinations of $Th_{\max}$ and $Th_{\min}$, which are described in **Table 1**.

Table 1. The combination of different parameters

Thmin \ Thmax	0.5	0.6	0.7	0.8	0.9
0.1	COM1	COM2	COM3	COM4	COM5
0.2	COM6	COM7	COM8	COM9	COM10
0.3	COM11	COM12	COM13	COM14	COM15
0.4	COM16	COM17	COM18	COM19	COM20

Table 2 shows the results of lists 20 different strategies. **Table 3** shows the Friedman tests and gives the values of ranking. The best combination is COM8. Therefore, the value of *Threshold* is $[0.2,\ 0.7]$, the performance of FTBA-TCR algorithm is the best.

Table 2. Comparision of different combination parameters

Func	COM 1	COM 2	COM 3	COM 4	COM 5	COM 6	COM 7
F1	6.8212×10^{-13}	2.2737×10^{-13}	2.2737×10^{-13}	2.2737×10^{-13}	2.2737×10^{-13}	2.2737×10^{-13}	2.2737×10^{-13}
F2	1.2690×10^{5}	1.4498×10^{5}	1.3795×10^{5}	1.5339×10^{5}	1.7784×10^{5}	1.4353×10^{5}	1.5637×10^{5}
F3	1.7749×10^{7}	2.1122×10^{7}	1.5592×10^{7}	2.1273×10^{7}	1.8244×10^{7}	1.6944×10^{7}	1.3784×10^{7}
F4	2.2153×10^{-4}	1.3726×10^{-4}	1.0489×10^{-4}	8.6563×10^{-5}	1.2155×10^{-4}	1.9956×10^{-4}	1.1390×10^{-4}
F5	2.0433×10^{-3}	2.3010×10^{-3}	2.3604×10^{-3}	2.4893×10^{-3}	2.6527×10^{-3}	2.1410×10^{-3}	2.3029×10^{-3}
F6	3.9890×10^{1}	3.8097×10^{1}	3.3494×10^{1}	3.3385×10^{1}	3.9907×10^{1}	2.9956×10^{1}	3.6306×10^{1}
F7	2.5157×10^{1}	2.9355×10^{1}	2.7921×10^{1}	2.3859×10^{1}	2.3214×10^{1}	2.3841×10^{1}	2.0637×10^{1}
F8	2.0898×10^{1}	2.0884×10^{1}	2.0894×10^{1}	2.0895×10^{1}	2.0885×10^{1}	2.0883×10^{1}	2.0898×10^{1}
F9	1.3791×10^{1}	1.4013×10^{1}	1.3967×10^{1}	1.3439×10^{1}	1.3503×10^{1}	1.4163×10^{1}	1.4413×10^{1}
F10	3.9378×10^{-2}	4.2156×10^{-2}	3.2559×10^{-2}	4.1597×10^{-2}	3.4435×10^{-2}	4.3865×10^{-2}	3.8932×10^{-2}
F11	6.3287×10^{1}	6.4145×10^{1}	6.5218×10^{1}	6.5375×10^{1}	6.5082×10^{1}	6.4848×10^{1}	6.1570×10^{1}
F12	6.2522×10^{1}	6.1512×10^{1}	6.0731×10^{1}	5.8449×10^{1}	5.8439×10^{1}	6.0936×10^{1}	6.1595×10^{1}
F13	1.1923×10^{2}	1.1836×10^{2}	1.2549×10^{2}	1.1908×10^{2}	1.2112×10^{2}	1.2019×10^{2}	1.1384×10^{2}
F14	2.9487×10^{3}	2.9222×10^{3}	2.9850×10^{3}	2.8085×10^{3}	2.9125×10^{3}	2.7503×10^{3}	2.9714×10^{3}
F15	2.9533×10^{3}	2.7016×10^{3}	2.7671×10^{3}	2.9430×10^{3}	2.8255×10^{3}	2.7777×10^{3}	2.7392×10^{3}
F16	1.5338×10^{-1}	1.4313×10^{-1}	1.6116×10^{-1}	1.3915×10^{-1}	1.4800×10^{-1}	1.3007×10^{-1}	1.5411×10^{-1}
F17	8.8667×10^{1}	8.9864×10^{1}	9.1630×10^{1}	9.1318×10^{1}	8.7573×10^{1}	9.0302×10^{1}	9.2847×10^{1}
F18	9.1893×10^{1}	8.4968×10^{1}	8.6819×10^{1}	9.3096×10^{1}	9.1011×10^{1}	8.6344×10^{1}	8.9386×10^{1}
F19	3.6787×10^{0}	3.8504×10^{0}	3.9958×10^{0}	4.0400×10^{0}	3.6871×10^{0}	3.6696×10^{0}	3.8538×10^{0}
F20	1.3766×10^{1}	1.3454×10^{1}	1.4015×10^{1}	1.3547×10^{1}	1.3479×10^{1}	1.3464×10^{1}	1.3864×10^{1}
F21	3.1980×10^{2}	3.1527×10^{2}	3.2201×10^{2}	3.0130×10^{2}	3.3020×10^{2}	2.9310×10^{2}	3.0266×10^{2}
F22	3.1485×10^{3}	3.1061×10^{3}	3.2697×10^{3}	3.0980×10^{3}	3.2423×10^{3}	3.2880×10^{3}	3.1858×10^{3}
F23	3.2440×10^{3}	2.9998×10^{3}	3.2000×10^{3}	3.1085×10^{3}	3.1991×10^{3}	3.2542×10^{3}	3.0179×10^{3}
F24	2.2848×10^{2}	2.2859×10^{2}	2.2855×10^{2}	2.3078×10^{2}	2.2833×10^{2}	2.3048×10^{2}	2.2797×10^{2}
F25	2.6325×10^{2}	2.6408×10^{2}	2.6430×10^{2}	2.6306×10^{2}	2.6744×10^{2}	2.6178×10^{2}	2.5829×10^{2}
F26	2.0001×10^{2}	2.0001×10^{2}	2.0001×10^{2}	2.0001×10^{2}	2.0001×10^{2}	2.0001×10^{2}	2.0001×10^{2}
F27	6.3509×10^{2}	6.4953×10^{2}	6.6936×10^{2}	6.3504×10^{2}	6.3668×10^{2}	6.6255×10^{2}	6.8195×10^{2}
F28	3.5741×10^{2}	3.5573×10^{2}	3.2068×10^{2}	3.4897×10^{2}	3.6293×10^{2}	3.5807×10^{2}	4.7485×10^{2}

Table 2. Comparision of different combination parameters (continue)

Func	COM 8	COM 9	COM 10	COM 11	COM 12	COM 13	COM 14
F1	2.2737×10^{-13}	2.2737×10^{-13}	2.2737×10^{-13}	2.2737×10^{-13}	2.2737×10^{-13}	2.2737×10^{-13}	2.2737×10^{-13}
F2	1.7306×10^{5}	1.7761×10^{5}	1.9607×10^{5}	1.6582×10^{5}	1.5637×10^{5}	1.8789×10^{5}	1.9613×10^{5}
F3	1.3429×10^{7}	2.1660×10^{7}	2.0369×10^{7}	8.1073×10^{6}	1.9927×10^{7}	1.1813×10^{7}	1.7697×10^{7}
F4	9.4016×10^{-5}	8.4178×10^{-5}	1.1124×10^{-4}	2.1803×10^{-4}	1.1569×10^{-4}	9.8091×10^{-5}	1.0063×10^{-4}
F5	2.5230×10^{-3}	2.4924×10^{-3}	2.7568×10^{-3}	2.1033×10^{-3}	2.2265×10^{-3}	2.4554×10^{-3}	2.5633×10^{-3}
F6	2.9437×10^{1}	4.1038×10^{1}	3.6857×10^{1}	3.1040×10^{1}	3.2432×10^{1}	3.5880×10^{1}	3.3704×10^{1}
F7	2.2642×10^{1}	2.4602×10^{1}	2.4438×10^{1}	2.4399×10^{1}	2.5299×10^{1}	1.7732×10^{1}	2.2445×10^{1}
F8	2.0875×10^{1}	2.0893×10^{1}	2.0911×10^{1}	2.0890×10^{1}	2.0890×10^{1}	2.0881×10^{1}	2.0891×10^{1}
F9	1.4582×10^{1}	1.3998×10^{1}	1.3545×10^{1}	1.4125×10^{1}	1.3470×10^{1}	1.4267×10^{1}	1.3463×10^{1}
F10	3.4446×10^{-2}	3.0337×10^{-2}	3.3413×10^{-2}	4.1012×10^{-2}	4.0287×10^{-2}	3.6133×10^{-2}	3.1212×10^{-2}
F11	6.0887×10^{1}	6.2195×10^{1}	6.5355×10^{1}	6.2429×10^{1}	6.1473×10^{1}	6.2682×10^{1}	6.8164×10^{1}
F12	5.9756×10^{1}	5.9646×10^{1}	5.9487×10^{1}	5.8923×10^{1}	5.8917×10^{1}	6.4048×10^{1}	6.2058×10^{1}
F13	1.1919×10^{2}	1.2023×10^{2}	1.2172×10^{2}	1.2061×10^{2}	1.2095×10^{2}	1.1870×10^{2}	1.2176×10^{2}
F14	2.8274×10^{3}	2.8499×10^{3}	2.8672×10^{3}	3.0040×10^{3}	3.0274×10^{3}	2.8986×10^{3}	2.9446×10^{3}
F15	2.8242×10^{3}	2.7284×10^{3}	2.7576×10^{3}	2.7714×10^{3}	2.8147×10^{3}	2.9104×10^{3}	2.7162×10^{3}
F16	1.3950×10^{-1}	1.4750×10^{-1}	1.2448×10^{-1}	1.3390×10^{-1}	1.4454×10^{-1}	1.2302×10^{-1}	1.3395×10^{-1}
F17	8.6528×10^{1}	9.1072×10^{1}	9.2330×10^{1}	8.9521×10^{1}	9.4067×10^{1}	9.1986×10^{1}	9.0026×10^{1}
F18	9.4002×10^{1}	8.9698×10^{1}	9.1778×10^{1}	9.2621×10^{1}	9.4252×10^{1}	8.9074×10^{1}	9.1406×10^{1}
F19	4.0525×10^{0}	4.0216×10^{0}	4.1881×10^{0}	4.1489×10^{0}	4.1139×10^{0}	4.0454×10^{0}	3.9727×10^{0}
F20	1.3261×10^{1}	1.3798×10^{1}	1.3878×10^{1}	1.3774×10^{1}	1.3538×10^{1}	1.4078×10^{1}	1.3709×10^{1}
F21	3.2065×10^{2}	3.0069×10^{2}	3.1758×10^{2}	3.2346×10^{2}	2.8245×10^{2}	3.0240×10^{2}	3.2065×10^{2}
F22	3.0847×10^{3}	3.2288×10^{3}	3.1061×10^{3}	3.0958×10^{3}	3.1501×10^{3}	3.1419×10^{3}	3.2268×10^{3}
F23	3.2131×10^{3}	3.2455×10^{3}	3.1905×10^{3}	3.3290×10^{3}	3.2146×10^{3}	3.1528×10^{3}	3.1470×10^{3}
F24	2.3005×10^{2}	2.2357×10^{2}	2.2728×10^{2}	2.3134×10^{2}	2.2614×10^{2}	2.3016×10^{2}	2.2928×10^{2}
F25	2.6612×10^{2}	2.6717×10^{2}	2.6670×10^{2}	2.6470×10^{2}	2.6817×10^{2}	2.6223×10^{2}	2.6266×10^{2}
F26	2.0244×10^{2}	2.0261×10^{2}	2.0281×10^{2}	2.0515×10^{2}	2.0997×10^{2}	2.0001×10^{2}	2.0267×10^{2}
F27	6.4332×10^{2}	6.5627×10^{2}	6.5866×10^{2}	6.6432×10^{2}	6.6827×10^{2}	6.5985×10^{2}	6.4319×10^{2}
F28	3.1920×10^{2}	3.8254×10^{2}	4.1812×10^{2}	3.2496×10^{2}	3.2888×10^{2}	3.2959×10^{2}	4.5869×10^{2}

Table 2. Comparision of different combination parameters (continue)

Func	COM15	COM16	COM17	COM18	COM19	COM20
F1	2.2737×10^{-13}	2.2737×10^{-13}	6.8212×10^{-13}	2.2737×10^{-13}	2.2737×10^{-13}	2.2737×10^{-13}
F2	2.3150×10^{5}	1.6988×10^{5}	1.7990×10^{5}	1.9360×10^{5}	1.9535×10^{5}	2.2924×10^{5}
F3	2.1156×10^{7}	1.6650×10^{7}	1.6425×10^{7}	2.4764×10^{7}	1.8991×10^{7}	1.9728×10^{7}
F4	9.7210×10^{-5}	1.6690×10^{-4}	9.9282×10^{-5}	1.1496×10^{-4}	8.9987×10^{-5}	1.2195×10^{-4}
F5	2.8344×10^{-3}	2.1110×10^{-3}	2.2791×10^{-3}	2.4425×10^{-3}	2.5202×10^{-3}	2.7608×10^{-3}
F6	3.7873×10^{1}	3.0784×10^{1}	3.7901×10^{1}	2.8876×10^{1}	3.8042×10^{1}	4.1538×10^{1}
F7	2.0936×10^{1}	2.1854×10^{1}	1.7995×10^{1}	2.2254×10^{1}	2.0554×10^{1}	2.1320×10^{1}
F8	2.0890×10^{1}	2.0908×10^{1}	2.0904×10^{1}	2.0892×10^{1}	2.0911×10^{1}	2.0906×10^{1}
F9	1.3861×10^{1}	1.3818×10^{1}	1.3659×10^{1}	1.4093×10^{1}	1.3816×10^{1}	1.3127×10^{1}
F10	2.8638×10^{-2}	4.2324×10^{-2}	3.4209×10^{-2}	2.9178×10^{-2}	2.6282×10^{-2}	2.4784×10^{-2}
F11	6.0985×10^{1}	5.9268×10^{1}	6.5687×10^{1}	6.1941×10^{1}	6.2916×10^{1}	6.3892×10^{1}
F12	5.7906×10^{1}	5.8605×10^{1}	6.1831×10^{1}	6.0572×10^{1}	6.4048×10^{1}	5.6186×10^{1}
F13	1.2368×10^{2}	1.2314×10^{2}	1.1850×10^{2}	1.2747×10^{2}	1.1430×10^{2}	1.1710×10^{2}
F14	2.7609×10^{3}	2.9245×10^{3}	2.8776×10^{3}	2.9420×10^{3}	2.9200×10^{3}	2.9133×10^{3}
F15	2.7506×10^{3}	2.7010×10^{3}	2.8407×10^{3}	2.8069×10^{3}	2.7651×10^{3}	2.7336×10^{3}
F16	1.4017×10^{-1}	1.2483×10^{-1}	1.3339×10^{-1}	1.4041×10^{-1}	1.2358×10^{-1}	1.5685×10^{-1}
F17	9.0008×10^{1}	9.3745×10^{1}	9.2036×10^{1}	9.1819×10^{1}	9.2385×10^{1}	8.9078×10^{1}
F18	9.4794×10^{1}	9.2682×10^{1}	8.8426×10^{1}	9.1333×10^{1}	8.9675×10^{1}	9.3375×10^{1}
F19	3.9869×10^{0}	3.9559×10^{0}	3.9799×10^{0}	3.7749×10^{0}	4.2087×10^{0}	3.9641×10^{0}
F20	1.3568×10^{1}	1.3428×10^{1}	1.3776×10^{1}	1.3162×10^{1}	1.3844×10^{1}	1.4101×10^{1}
F21	3.1246×10^{2}	3.0632×10^{2}	3.0291×10^{2}	3.0743×10^{2}	3.1502×10^{2}	3.2261×10^{2}
F22	3.2526×10^{3}	3.0983×10^{3}	3.0729×10^{3}	3.3560×10^{3}	3.1408×10^{3}	3.1063×10^{3}
F23	3.2302×10^{3}	3.2893×10^{3}	3.2463×10^{3}	3.1797×10^{3}	3.1078×10^{3}	3.2236×10^{3}
F24	2.2938×10^{2}	2.2764×10^{2}	2.3342×10^{2}	2.3030×10^{2}	2.2677×10^{2}	2.2968×10^{2}
F25	2.6800×10^{2}	2.6835×10^{2}	2.6235×10^{2}	2.6519×10^{2}	2.6177×10^{2}	2.5929×10^{2}
F26	2.0290×10^{2}	2.1037×10^{2}	2.1054×10^{2}	2.0234×10^{2}	2.0780×10^{2}	2.0262×10^{2}
F27	6.6289×10^{2}	6.4500×10^{2}	6.9046×10^{2}	6.5858×10^{2}	6.5364×10^{2}	6.6596×10^{2}
F28	3.6756×10^{2}	3.9782×10^{2}	3.4290×10^{2}	3.5396×10^{2}	3.7486×10^{2}	3.6061×10^{2}

Table 3. Algorithm ranking corresponding to different combination

Thmin \ Thmax	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
0.1	11.07	9.55	11.50	9.04	10.66
0.2	9.46	9.88	**9.02**	10.79	11.79
0.3	11.52	11.41	9.71	10.89	11.29
0.4	9.93	10.79	10.75	10.02	10.95

4.2 Comparison of FTBA-TCR with other algorithms

To further test the performance of FTBA-TCR, the other four algorithms are compared in CEC2013 test set. The involved algothrims are listed as follows:

(1) Bat Algorithm (BA) [46]

(2) Particle Swarm Optimizer (PSO) [48]

(3) Fast Triangle Flip Bat Algorithm (FTBA) [47]

(4) Fast Triangle Flip Bat Algorithm with Curve Strategy (FTBA-TC) [39]

Table 4 presents the values of mean error function which are achieved by the four algorithms on the CEC2013 test suite. And the results show that FTBA-TCR has the best performance. Specifically, FTBA-TCR performs better than BA on 27 functions. For the PSO and FTBA, FTBA-TCR performs better on 19 functions and 21 functions, respecively. While PSO and FTBA outperform FTBA-TCR on 6 functions and 4 functions, respectively. Meanwhile, compared to FTBA-TC, FTBA-TCR obtains better results on 16 functions, and loses in 7 functions.

Table 4. Comparison results of FTBA-TCR with other algorithms (D = 30)

Func	BA	PSO	FTBA	FTBA-TC	FTBA-TCR
F1	2.1746×10^{0}	9.8908×10^{-11}	9.6300×10^{-5}	1.0732×10^{-10}	$\mathbf{2.2737\times10^{-13}}$
F2	3.4281×10^{6}	1.8621×10^{7}	$\mathbf{4.2191\times10^{4}}$	7.0320×10^{4}	1.7306×10^{5}
F3	3.4836×10^{8}	4.7707×10^{9}	$\mathbf{7.7470\times10^{6}}$	1.2630×10^{7}	1.3429×10^{7}
F4	3.3322×10^{4}	1.9453×10^{4}	8.8985×10^{-2}	1.4314×10^{-1}	$\mathbf{9.4016\times10^{-5}}$
F5	5.9588×10^{-1}	$\mathbf{5.7260\times10^{-6}}$	1.1141×10^{-3}	1.8417×10^{-3}	2.5230×10^{-3}
F6	5.9872×10^{1}	2.5047×10^{6}	$\mathbf{6.7154\times10^{0}}$	3.0459×10^{1}	2.9437×10^{1}
F7	3.5259×10^{2}	9.9763×10^{1}	1.0567×10^{2}	3.0450×10^{1}	$\mathbf{2.2642\times10^{1}}$
F8	2.0946×10^{1}	2.0912×10^{1}	2.0943×10^{1}	2.0891×10^{1}	$\mathbf{2.0875\times10^{1}}$
F9	3.5449×10^{1}	2.8324×10^{1}	2.9771×10^{1}	1.5559×10^{1}	$\mathbf{1.4582\times10^{1}}$
F10	1.3552×10^{0}	2.5047×10^{1}	7.7905×10^{-2}	4.9368×10^{-2}	$\mathbf{3.4446\times10^{-2}}$
F11	4.4855×10^{2}	$\mathbf{3.6165\times10^{1}}$	3.4451×10^{2}	6.8145×10^{1}	6.0887×10^{1}
F12	4.3674×10^{2}	1.2252×10^{2}	3.5205×10^{2}	6.4106×10^{1}	$\mathbf{5.9756\times10^{1}}$
F13	4.5750×10^{2}	2.1430×10^{2}	3.5770×10^{2}	1.2552×10^{2}	$\mathbf{1.1919\times10^{2}}$
F14	4.8155×10^{3}	$\mathbf{9.0358\times10^{2}}$	4.2722×10^{3}	2.9522×10^{3}	2.8274×10^{3}

F15	4.8636×10^{3}	6.9169×10^{3}	4.4365×10^{3}	3.0913×10^{3}	$\mathbf{2.8242\times10^{3}}$
F16	2.1238×10^{0}	2.3457×10^{0}	3.4991×10^{-1}	1.7844×10^{-1}	$\mathbf{1.3950\times10^{-1}}$
F17	9.2805×10^{2}	$\mathbf{6.8637\times10^{1}}$	1.7236×10^{2}	9.0855×10^{1}	8.6528×10^{1}
F18	9.5058×10^{2}	2.3090×10^{2}	$\mathbf{1.4139}\times10^{2}$	$\mathbf{9.0476\times10^{1}}$	9.4002×10^{1}
F19	5.7147×10^{1}	1.6124×10^{1}	7.1370×10^{0}	$\mathbf{3.9900\times10^{0}}$	4.0525×10^{0}
F20	1.4541×10^{1}	1.2931×10^{1}	1.2588×10^{1}	$\mathbf{1.0430\times10^{1}}$	1.3261×10^{1}
F21	$\mathbf{3.1531\times10^{2}}$	3.3438×10^{2}	3.3336×10^{2}	3.1673×10^{2}	3.2065×10^{2}
F22	6.0955×10^{3}	$\mathbf{7.2620\times10^{2}}$	5.4886×10^{3}	3.1905×10^{3}	3.0847×10^{3}
F23	5.8443×10^{3}	7.4071×10^{3}	5.5490×10^{3}	3.2542×10^{3}	$\mathbf{3.2131\times10^{3}}$
F24	3.2093×10^{2}	2.8232×10^{2}	2.9483×10^{2}	2.3198×10^{2}	$\mathbf{2.3005\times10^{2}}$
F25	3.5239×10^{2}	3.2577×10^{2}	3.2681×10^{2}	2.7072×10^{2}	$\mathbf{2.6612\times10^{2}}$
F26	2.0776×10^{2}	2.0026×10^{2}	$\mathbf{2.0354}\times10^{2}$	$\mathbf{2.0000\times10^{2}}$	2.0244×10^{2}
F27	1.3088×10^{3}	9.3485×10^{2}	1.1271×10^{3}	7.0631×10^{2}	$\mathbf{6.4332\times10^{2}}$
F28	3.7436×10^{3}	3.2071×10^{2}	2.2526×10^{3}	3.5307×10^{2}	$\mathbf{3.1920\times10^{2}}$
w/t/l	27/0/1	19/3/6	21/3/4	16/5/7	

4.3 DV-Hop Localization Algorithm with Different Strategies

We conducted simulation experiments in MATLAB R2013a to test and verify the performance of the proposed algorithm. And the parameters are set as follows in the simulation environment:

Table 5. Parameter values

Parameter	Value
Network Zones	100m×100m
Population Size	10
Iterations	60
Communication Radius, CR (m)	25
Number of Nodes	100
Beacon nodes	20
Max-Hop	5

Average localization error was used to evalute the localization performance:

$$Average\ error = \frac{100}{n\times R}\sum_{i=1}^{n}\sqrt{(x_i^{'}-x_i)^2+(y_i^{'}-y_i)^2} \tag{22}$$

where, the number of ordinary nodes is n, R is CR, $(x_i^{'}, y_i^{'})$ is the estimated location of ordinary nodes, and (x_i, y_i) is the real location of ordinary nodes.

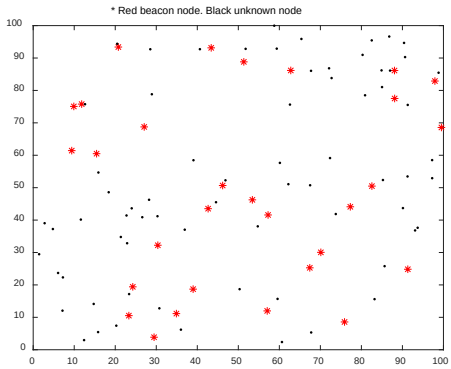

Fig. 3. Initial diagram of the node's location

The following algorithms are applied in DV-Hop algorithm to decrease the average localization error.

a) DV-Hop algorithm, shorthand for DVHop algorithm
b) DV-Hop algorithm based on PSO which is shorthand for PSO-DVHop
c) DV-Hop algorithm based on Standard-BA which is shorthand for BA-DVHop
d) DV-Hop algorithm based on FTBA which is shorthand for FTBA-DVHop
e) DV-Hop algorithm based on FTBA-TC which is shorthand for FTBA-TC-DVHop
f) DV-Hop algorithm based on FTBA-TCR which is shorthand for FTBA-TCR-DVHop

(1) The changes in communication radius (CR)

Table 6 and **Fig. 4** show the influence of communication radius change on algorithm performance. Obviously, with the increase of CR, the error decreases gradually, and FTBA-TCR-DVHop has the best performance.

Table 6. Comparison of average localization errors of different CR

Communication radius R(m)	**15**	**20**	**25**	**30**	**35**	**40**
DVHop	65.2355	46.1420	33.2461	28.9185	27.5943	26.5377
PSO-DVHop	61.3735	27.4250	26.2677	22.0896	20.5442	18.2681
BA-DVHop	134.8861	109.8963	67.0698	45.5068	36.0811	35.0503
FTBA-DVHop	61.9326	27.9955	27.2166	22.5006	20.6753	18.6560
FTBA-TC-DVHop	61.0395	31.5839	26.0766	**21.8039**	20.5630	18.5417
FTBA-TCR-DVHop	**60.3637**	**26.7855**	**25.9177**	21.9265	**20.3831**	**18.1419**

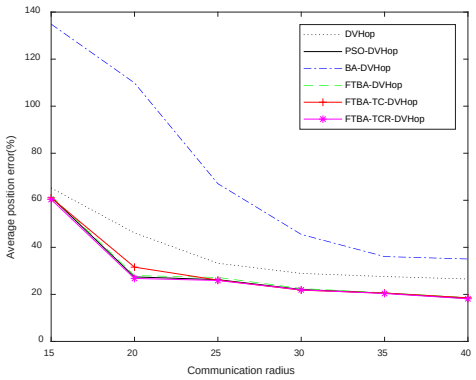

Fig. 4. Comparison of average localization errors of different CR

(2) The changes in the total number of nodes

Fig. 5 and **Table 7** describe the change of algorithm performance on the condition that the total number of nodes changes while beacon nodes remains unchanged. The smaller the number of nodes, the larger the average distance error per hop, indicating that the error decreases with the increase of the number of nodes. And FTBA-TCR-DVHop algorithm performance is always better than the other algorithms.

Table 7. Comparison of average localization errors of different node numbers

Number of nodes	50	60	70	80	90	100
DV-Hop	51.7015	43.6030	30.5574	32.5681	33.1297	33.2461
PSO-DVHop	45.8702	30.2964	26.9166	26.3520	27.3228	26.2677
BA-DVHop	100.8718	82.7320	80.7297	67.2410	73.6408	67.0698
FTBA	46.0046	31.3394	28.1952	26.5801	27.9941	27.2166
FTBA-TC-DVHop	45.6892	30.2959	26.8246	26.2629	27.1237	26.0766
FTBA-TCR-DVHop	**45.6449**	**30.2843**	**26.6181**	**25.7657**	**27.1213**	**25.9177**

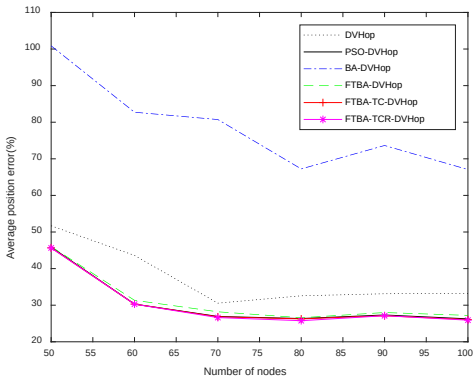

Fig. 5. Comparison of average localization errors of different node numbers

(3) The changes in the number of beacon nodes

Fig. 6 and **Table 8** shows the impact of beacon nodes changes on algorithm performance. When there are fewer beacon nodes (e.g., 5), the performance of DVHop algorithm based on intelligent optimization algorithm is worse than DVHop algorithm, but with the change of the amount of beacon nodes, the performance of DVHop algorithm based on intelligent optimization algorithm becomes better than DVHop algorithm. And in most case, the FTBA-TCR-DVHop has the minimum position error.

Table 8. Comparison of average localization errors of different beacon nodes

Number of beacon nodes	5	10	15	20	25	30
DVHop	**49.2066**	38.2098	38.7743	33.2461	28.3095	32.4829
PSO-DVHop	56.0591	33.8158	31.9287	26.2677	23.6661	21.5371
BA-DVHop	83.6027	64.2895	70.1913	67.0698	52.3613	54.0451
FTBA	57.7668	34.1958	32.3871	27.2166	23.6249	21.4006
FTBA-TC-DVHop	55.6318	**33.4534**	31.3620	26.0766	**23.3446**	21.6131
FTBA-TCR-DVHop	55.5642	33.7952	**31.2128**	**25.9177**	23.5206	**21.3394**

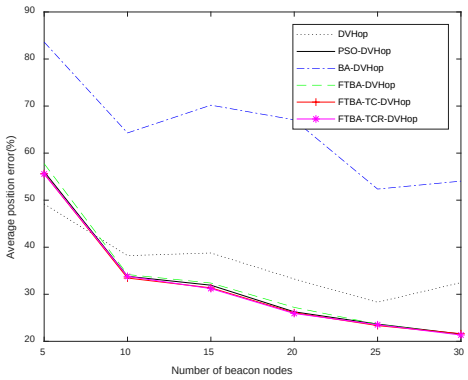

Fig. 6. Comparison of average localization errors of different beacon nodes

(4) The number of iterations is different

Fig. 7 and **Table 9** describes the change in the number of with the times of iterations will affect the performance of the DV-Hop algorithm. Obviously, with the increase of iterations, the performance of DVHop algorithm based on intelligent optimization algorithm becomes better. What's more, FTBA-TCR-DVHop converges faster than other algorithms, which has better converges performance than other algorithms.

Table 9. Comparison of average localization errors of different iterations

Iteration times	10	20	30	40	50	60	70
BA-DVHop	139.8698	108.0632	107.7151	99.2140	81.5134	67.0698	60.9370
PSO-DVHop	52.9496	33.8195	26.9679	26.1860	26.4091	26.2677	26.1135
FTBA-DVHop	82.3594	34.6621	28.1647	27.3993	27.2004	27.2166	26.5344
FTBA-TC-DVHop	61.7504	37.6602	29.8776	27.4943	26.4402	26.0766	25.9309
FTBA-TCR-DVHop	**49.9785**	**28.4519**	**26.7170**	**25.9629**	**25.9510**	**25.9177**	**25.8849**

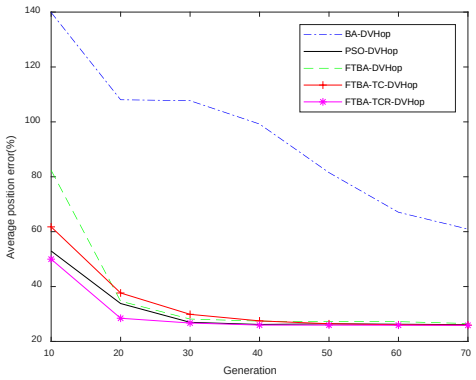

Fig. 7. Comparison of average localization errors of different iterations

5. Conclusion

DV-Hop is a general range-free positioning algorithm for detecting ordinary nodes position. Because of the simple location principle of DV-Hop algorithm, it brings a higher localization error. In this paper, FTBA-TCR is designed to modify the accuracy of DV-Hop. In FTBA-TCR, the local search ability is affected by the optimal location. In each generation, the rank-based transformation strategy is used to select which individuals to conduct a local search. It was tested in the CEC2013 benchmark and applied into the DV-Hop algorithm. The best combination of threshold is adopted by the CEC2013 test suite. Four different algorithms are used to compare with the proposed algorithm. And the DV-Hop algorithms base these intelligent algorithms are used to verify the performance of FTBA-TCR. The simulation results show that the FTBA-TCR strategy is added to the localization of wireless sensor nodes which achieve better localization performance. In the future, we will integrate other operators and coupling strategies to improve the performance of BA. In addition, the proposed algorithm can be applied for solving practical problems, such as: the vehicle routing problem (VRP) [49], integer programming problems [50], the numerical association rule mining problem [51].

Acknowledgements

This work is supported by the National Natural Science Foundation of China under Grant No.61806138, Natural Science Foundation of Shanxi Province under Grant No.201801D121127, Taiyuan University of Science and Technology Scientific Research Initial Funding under Grant No.20182002.

References

[1] K. Akkaya and M. Younis, "A survey on routing protocols for wireless sensor networks," *Ad hoc networks*, vol. 3, no. 3, pp. 325-349, 2005. Article (CrossRef Link)

[2] Z. Z. Tang, H. Y. Wang, Q. Hu and H. Long. "How Network Coding Benefits Converge-Cast in Wireless Sensor Networks," *KSII Transactions on Internet & Information Systems*, vol. 7, no. 5, pp. 1180-1197, 2013. Article (CrossRef Link)

[3] C. Chow, M. Mokbel and T. He, "A privacy-preserving location monitoring system for wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 10, no. 1, pp. 94-107, 2011. Article (CrossRef Link)

[4] Z. B. Wang, "Node localization of wireless sensor networks based on bat algorithm," *Comp Eng & Appl*, vol. 50, no. 11, pp. 90-94, 2014. Article (CrossRef Link)
[5] A. M. Hegland, E. Winjum, S. F. Mjolsnes, C. Rong, O. Kure and P. Spilling, "A survey of key management in ad hoc networks," *Communications Surveys & Tutorials IEEE*, vol. 8, no.3, pp. 48-66, 2006. Article (CrossRef Link)
[6] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, "A survey on sensor networks," *IEEE communications magazine*, vol. 40, no 8, pp. 102-114, 2002. Article (CrossRef Link)
[7] P. Guo, J. Wang, X. H. Geng, C. S. Kim and J. U. Kim, "A variable threshold-value authentication architecture for wireless mesh networks," *Internet Technol*, vol. 15, no. 6, pp. 929-935, 2014. Article (CrossRef Link)
[8] S. Phoemphon, C. So-In, and N. Leelathakul, "Optimized Hop Angle Relativity for DV-Hop Localization in Wireless Sensor Networks," *IEEE Access*, vol. 6, pp. 78149-78172, 2018. Article (CrossRef Link)
[9] A. Harter, A. Hopper, P. Steggles, A. Ward and P. Webster, "The Anatomy of a Context-Aware Application," *the Journal of Mobile Communication, Computation and Information*, vol. 8, pp. 187-197, 2002. Article (CrossRef Link)
[10] D. Niculescu, D. Lab, and B. Nath, "Ad hoc positioning system (APS) using AoA," in *Proc. of 22nd Annual Joint Conference of the IEEE Computer and Communications*, pp. 1734-1743, March 30- April 3, 2003. Article (CrossRef Link)
[11] L. Girod, V. Bychkovskiy, J. Elson and D. Estrin, "Locating tiny sensors in time and space: a case study," in *Proc. of International Conference on Computer Design VLSI in Copmuters and Processors*, pp. 214-219, September 16-18, 2002. Article (CrossRef Link)
[12] D. Niculescu, and B. Nath, "DV based positioning in ad hoc networks," *Telecommunication Systems*, vol. 22 no. 1-4, pp. 267-280, 2003. Article (CrossRef Link)
[13] L. Doherty, K. S. J. Pister, and L. El Ghaoui, "Convex position estimation in wireless sensor networks," in *Proc. of 20th Annual Joint Conference of the IEEE Computer and Communications Societies*, pp. 1655-1663, April 24-26, 2001. Article (CrossRef Link)
[14] C. H. Wu, W. H. Sheng and Y. Zhang, "Mobile Sensor Networks Self Localization based on Multi-dimensional Scaling," in *Proc. of IEEE International Conference on Robotics and Automation*, pp. 4038-4043, April 10-14, 2007. Article (CrossRef Link)
[15] Z. Wei, S. Su, and S. Fei, "Improved DV-Hop Algorithm Using Locally Weighted Linear Regression in Anisotropic Wireless Sensor Networks," *Wireless Personal Communications*, vol. 98, no. 4, pp. 3335-3353, 2018. Article (CrossRef Link)
[16] L. Kezhong, Y. Xinping and H. Fuping, "A modified DV-Hop localization algorithm for wireless sensor networks," in *Proc. of IEEE International Conference on Intelligent Computing and Intelligent Systems*, pp. 511-514, November 20-22, 2009. Article (CrossRef Link)
[17] Z. H. Cui, Y. Chang, J. J. Zhang, X. J. Cai and W. S. Zhang, "Improved NSGA-III with selection-and-elimination operator," *Swarm and Evolutionary Computataion*, vol. 49, pp. 22-23, 2019. Article (CrossRef Link)
[18] R. C. He, C. X. Ma, X. Y. Jia, Q. Xiao and L. Qi, "Optimisation of dangerous goods transport based on the improved ant colony algorithm," *International Journal of Computing Science and Mathematics*, vol. 8, no. 3, pp. 210-217, 2017. Article (CrossRef Link)
[19] B. P. Zhao, Y. Xue, B. Xu, T. H. Ma and J. F. Liu, "Multi-objective classification based on NSGA-II," *International Journal of Computing Science and Mathematics*, vol. 9, no. 6, pp. 539-546. 2018. Article (CrossRef Link)
[20] Z. H. Cui, L. Du, P. H. Wang, X. J. Cai and W. S. Zhang, "Malicious code detection based on CNNs and multi-objective algorithm," *Journal of Parallel and Distributed Computing*, vol. 129, pp. 50-58, 2019. Article (CrossRef Link)
[21] Z. H. Cui, Y. Cao, X. J. Cai, J. H. Cai, and J. J. Chen, "Optimal LEACH protocol with modified bat algorithm for big data sensing systems in Internet of Things," *Journal of Parallel and Distributed Computing*, vol. 132, pp. 217-229, 2017. Article (CrossRef Link)

[22] J. C. Fan, Y. Li, L. Y. Tang and G. K. Wu, "RoughPSO: rough set-based particle swarm optimisation," *International Journal of Bio-Inspired Computation*, vol. 12, no. 4, pp. 245-253, 2018. Article (CrossRef Link)
[23] H. Y. Gao, Y. A. Du and M. Diao, "Quantum-inspired glowworm swarm optimisation and its application,"*International Journal of Computing Science and Mathematics*, vol. 8, no. 1, pp. 91 – 100, 2017. Article (CrossRef Link)
[24] M. Q. Zhang, H. Wang, Z. H. Cui, and J. J. Chen, "Hybrid multi-objective cuckoo search with dynamical local search," *Memetic Computing*, vol. 10, no. 2, pp. 199-208, 2018. Article (CrossRef Link)
[25] Z. H. Cui, J. J. Zhang, Y. C. Wang, Y. Cao, X. J. Cai, W. S. Zhang and J. J. Chen, "A pigeon-inspired optimization algorithm for many-objective optimization problems," *SCIENCE CHINA Information Sciences*, vol.62, no.7, pp. 70212, 2019. Article (CrossRef Link)
[26] E. Raj, and L. Babu, "A firefly inspired game dissemination and QoS-based priority pricing strategy for online social network games," *International Journal of Bio-Inspired Computation*, vol. 11, no. 3, pp. 202-217, 2018. Article (CrossRef Link)
[27] J. J. Zhang, F. Xue, X. J. Cai, Z. H. Cui, Y. Chang, W. S. Zhang and W. Z. Li, "Privacy protection based on many-objective optimization algorithm," *Concurrency and Computation Practice and Experience*, vol. 31, no. 20, 2019. Article (CrossRef Link)
[28] M. Abdel-Baset, Y. Q. Zhou and M. Ismail, "An improved cuckoo search algorithm for integer programming problems," *International Journal of Computing Science and Mathematics*, vol. 9, no. 1, pp. 66-81, 2018. Article (CrossRef Link)
[29] W. X. Yu and J. N. Wang, "A new method to solve optimisation problems via fixed point of firefly algorithm," *International Journal of Bio-Inspired Computation*, vol. 11, no. 4, pp. 249-256, 2018. Article (CrossRef Link)
[30] Z. H. Cui, M. Q. Zhang, H. Wang and W. S. Zhang, "A hybrid many-objective cuckoo search algorithm," *soft computing*, vol. 23, pp. 10681-10697, 2019. Article (CrossRef Link)
[31] Z. H. Cui, F. Xue, X. J. Cai, Y. Cao, G. G. Wang, and J. J. Chen, "Detection of Malicious Code Variants Based on Deep Learning," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 7, pp. 3187-3196, 2018. Article (CrossRef Link)
[32] Y. Cao, Z. M. Ding, F. Xue, X. T. Rong, "An improved twin support vector machine based on multi-objective cuckoo search for software defect prediction," *International Journal of Bio-Inspired Computation*, vol. 11, no. 4, pp. 282-291, 2018. Article (CrossRef Link)
[33] G. Zhou, R. X. Zhao, and Y. Q. Zhou, "Solving large-scale 0-1 knapsack problem by the social-spider optimisation algorithm," *International Journal of Computing Science and Mathematics*, vol. 9, no. 5, pp. 433-441, 2018. Article (CrossRef Link)
[34] M. Bougherara, N. Nedjah, L. D. M. Mourelle, R. Rahmoun, A. Sadok and D. Bennouar, "IP assignment for efficient NoC-based system design using multi-objective particle swarm optimization," *International Journal of Bio-Inspired Computation*, vol. 12, no. 4, pp. 203-213, 2018. Article (CrossRef Link)
[35] Z. H. Cui, B. Sun, G. G. Wang, Y. Xue, and J. J. Chen, "A novel oriented cuckoo search algorithm to improve DV-Hop performance for cyber–physical systems," *Journal of Parallel and Distributed Computing*, vol. 103, pp. 42-52, 2017. Article (CrossRef Link)
[36] G. Saranya, H. Nehemiah, and A. Kannan, "Hybrid particle swarm optimisation with mutation for code smell detection," *International Journal of Bio-Inspired Computation*, vol. 12, no. 3, pp. 186-195, 2018. Article (CrossRef Link)
[37] X. J. Cai, X. Z. Gao, and Y. Xue, "Improved bat algorithm with optimal forage strategy and random disturbance strategy," *International Journal of Bio-inspired Computation*, vol. 8, no. 4, pp. 205-214, 2016. Article (CrossRef Link)
[38] Z. H. Cui, J. J. Zhang, Y. C. Wang, Y. Cao, X. J. Cai, W. S. Zhang and J. J. Chen, "A pigeon-inspired optimization algorithm for many-objective optimization problems," *SCIENCE CHINA Information Sciences*, vol.62, no.7, pp. 70212, 2019. Article (CrossRef Link)

[39] X. J. Cai, Y. Sun, and Z. H. Cui, "Optimal LEACH Protocol with Improved Bat Algorithm in Wireless Sensor Networks," *KSII Transcactions on Internet and Information Systems*, vol. 13, no. 5, pp. 2469-2490, 2019. Article (CrossRef Link)
[40] G. Anastasi, M. Conti, M. D. Francesco, and A. Passarella, "Energy Conservation in Wireless Sensor Networks: A Survey," *Ad Hoc Networks*, vol. 7, no. 3, pp. 537-568, 2009. Article (CrossRef Link)
[41] C. Li, M. Yong, T. Li, L. Wei and Z. Zhao, "Overview of Wireless Sensor Networks," *Journal of Computer Research & Development*, vol. 36, no. 2, pp. 1-22, 2005.
[42] L. Gui, T. Val, A. Wei and R. Dalce, "Improvement of range-free localization technology by a novel DV-hop protocol in wireless sensor networks," *Ad Hoc Networks*, vol. 24, pp. 55-73, 2015. Article (CrossRef Link)
[43] R. Mohammadi, R. Javidan and M. Keshtgari, "An intelligent traffic engineering method for video surveillance systems over software defined networks using ant colony optimization," *International Journal of Bio-Inspired Computation*, vol. 12, no. 3, pp. 173-185, 2018. Article (CrossRef Link)
[44] Z. H. Cui, F. X. Li, and W. S. Zhang, "Bat algorithm with principal component analysis," *International Journal of Machine Learning and Cybernetics*, vol. 10, no. 3, pp. 603-622, 2019. Article (CrossRef Link)
[45] Gaige Wang, Xingjuan Cai, Zhihua Cui, Geyong Min and Jinjun Chen, "High Performance Computing for Cyber Physical Social Systems by Using Evolutionary Multi-Objective Optimization Algorithm," *IEEE Transactions on Emerging Topics in Computing*, pp. 1-1, 2017. Article (CrossRef Link)
[46] X. S. Yang, "A new metaheuristic bat-inspired algorithm," *Computer Knowledge & Technology*, vol. 284, pp. 65-74, 2010. Article (CrossRef Link)
[47] X. J. Cai, H. Wang, Z. H. Cui, J. H. Cai, Y. Xue, and L. Wang, "Bat algorithm with triangle-flipping strategy for numerical optimization," *International Journal of Machine Learning and Cybernetics*, vol. 9, no. 2, pp. 199-215, 2018. Article (CrossRef Link)
[48] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proc. of 3rd IEEE International Conference on Evolutionary Computation*, pp. 69-73, May 4-9, 1998. Article (CrossRef Link)
[49] T. Yigit, O. Unsal and O. Deperlioglu, "Using the metaheuristic methods for real-time optimisation of dynamic school bus routing problem and an application," *International Journal of Bio-Inspired Computation*, vol. 11, no. 2, pp. 123-133, 2018. Article (CrossRef Link)
[50] M. Abdel-Baset, Y. Q. Zhou and M. Ismail, "An improved cuckoo search algorithm for integer programming problems," *International Journal of Computing Science and Mathematics*, vol. 9, no. 1, pp. 66-81, 2018. Article (CrossRef Link)
[51] K. E. Heraguemi, N. Kamel and H. Drias, "Multi-objective bat algorithm for mining numerical association rules," *International Journal of Bio-Inspired Computation*, vol. 11, no. 4, pp. 239-248, 2018. Article (CrossRef Link)

Xingjuan Cai received her Ph.D. degree in Control Science and Engineering from Tongji University, China, in 2017. She is an Associate Professor with the school of Computer Science and Technology, Taiyuan University of Science and Technology, Taiyuan, China. Her research interest includes bio-inspired computation and application.

Shaojin Geng is currently working toward M.S. degree at computer science and technology, Taiyuan University of Science and Technology, Taiyuan, China. His research interest includes computational intelligence and combinatorial optimization.

Penghong Wang is currently working toward M.S. degree at computer science and technology, Taiyuan University of Science and Technology, Taiyuan, China. His main research includes computational intelligence and combinatorial optimization.

Lei Wang received his B.S. degree in Electrical Technology from the Jiangsu Institute of Technology, China, in 1992. He received his M.S. and Ph.D. degrees in Automation from the Tongji University, in 1995 and 1998, respectively. He is currently a professor at the Department of Control Science and Engineering, Tongji University. His research interest covers intelligent automation, computational intelligence, heuristic algorithm, system engineering.

Qidi Wu received her B.S. degree in Radio Technology, and M.S. degree in Automatic Control from the Tsinghua University, China, in 1970 and 1981, respectively. She received her Ph.D. degree in Automation from ETHZ, Zurich, Switzerland. She is currently a professor at the Department of Control Science and Engineering, Tongji University. Her research interest covers control theory and engineering, intelligent automation, heuristic algorithm, complex systems scheduling and optimization, system engineering and management engineering, smart scheduling of home energy management system.