Research Article

Distributed Beacon Drifting Detection for Localization in Unstable Environments

Ming Xia,[1] Peiliang Sun,[2] Xiaoyan Wang,[1] Yan Jin,[1] and Qingzhang Chen[1]

[1] *College of Computer Science and Technology, Zhejiang University of Technology, 288 Liuhe Road, Hangzhou 310023, China*
[2] *Department of IT and Management, Zhejiang Police Vocational Academy, No. 2 Street, Hangzhou 310018, China*

Correspondence should be addressed to Ming Xia; xiaming@zjut.edu.cn

Localization is a fundamental research issue in wireless sensor networks (WSNs). In most existing localization schemes, several beacons are used to determine the locations of sensor nodes. These localization mechanisms are frequently based on an assumption that the locations of beacons are known. Nevertheless, for many WSN systems deployed in unstable environments, beacons may be moved unexpectedly; that is, beacons are drifting, and their location information will no longer be reliable. As a result, the accuracy of localization will be greatly affected. In this paper, we propose a distributed beacon drifting detection algorithm to locate those accidentally moved beacons. In the proposed algorithm, we designed both beacon self-scoring and beacon-to-beacon negotiation mechanisms to improve detection accuracy while keeping the algorithm lightweight. Experimental results show that the algorithm achieves its designed goals.

1. Introduction

A wireless sensor network (WSN) [1, 2], which consists of a large number of sensor nodes to sense, process, and transmit monitored information in an environment cooperatively, has changed the way people interact with the physical world. Currently, WSN technologies have been widely adopted in environmental monitoring systems [3] and are foreseen to be used in more application fields, such as civil engineering [4] and sustainable energy systems [5]. Localization [6] is one of the most essential research issues in WSNs because the position of the monitored information is frequently one of our major concerns. Currently available localization algorithms typically use a small set of beacon nodes (or simply beacons) which are aware of their positions to deduce sensor nodes' positions through trilateration [7] or other techniques such as maximum likelihood estimation [8]. To mark the positions of beacons, we can either equip beacons with GPS receivers to let them calculate their locations automatically or mark beacons' positions manually in deployment. Because of relatively high price and power consumption, GPS receivers may not be available for many low-power WSN systems, and the second method is frequently used. As a result, these

algorithms are frequently based on an assumption that the beacons should be immobile and their position information is accurate. Nevertheless, this assumption does not hold in unstable environments. Because nodes in WSNs are typically battery powered [9], we frequently deploy WSN systems in unstable environments to perform monitoring tasks, and beacons' positions may be changed unexpectedly in this case. For instance, topographic change can move one or more beacons to other positions without being noticed. We denote these unexpected movements of beacons as that the beacons are drifting. The beacon drifting problem we discuss here is different from the well-investigated mobile beacon problem [10]. In the mobile beacon problem, beacons are moving along specific tracks in the monitored area to reduce the number of beacons required or to increase localization accuracy. Therefore, the position information of beacons is still accurate. But in the beacon drifting problem, beacons are unaware of the accidental movements, and the preset position information might be totally incorrect. In this occasion, the localization result will be seriously affected if we use inaccurate beacon position information.

In this paper, we focus on the beacon drifting detection problem, in which we have to determine: (1) whether all

beacons are still at their original deployed positions; (2) if not, which beacons have moved unexpectely. With this information, we can, for example, avoid using moved beacons in localization to ensure localization accuracy. In this paper, we introduce a distributed beacon drifting detection algorithm. We chose to let each beacon keep monitoring the variance of RSSI (received signal strength indication) measurement value between itself and its neighboring beacons and the beacon can use the monitored RSSI information to determine if itself has moved. We based our detection algorithm on RSSI measurement because it is available in almost all RF receivers on the market. However, the unreliable nature of RSSI measurement will cause serious fault detection problem. As a result, we first analyzed different types of fault detections and then designed beacon self-scoring and beacon-to-beacon negotiation mechanisms to reduce the possibility of fault detection while keeping low operating overhead.

The rest of the paper is organized as follows. Section 2 introduces the state of the art. Section 3 proposes the design of the distributed beacon drifting detection algorithm. Section 4 provides evaluation results, and Section 5 concludes the paper.

2. Related Work

Localization algorithms can be roughly categorized into range-based and range-free algorithms. Range-based algorithms require the distance information between beacons and sensor nodes in deducing the sensor nodes' positions. Currently available range-based algorithms frequently use RSSI [11] or TOA/TDOA (time of arrival/time difference of arrival) [12, 13] to estimate the distance. Range-free algorithms do not need distance information and alternatively use network connectivity information, such as hop counts [14] from beacons to sensor nodes, to estimate the locations of sensor nodes. Most currently available range-based and range-free algorithms are based on one essential assumption: beacons are immobile, and their position information is known and accurate.

Some researchers have proposed a different solution called "mobile beacon" for localization, in which one or more beacons are travelling along specific tracks in the monitored area [10, 15]. We can estimate one sensor node's position according to the distance measurement from the sensor node to the beacons, and the position information of beacons at that time. Through this "mobile beacon" method, it is possible for us to improve the accuracy of localization and reduce the number of beacons in the network. In these algorithms, the position information and moving tracks of beacons must be known and accurate. In this paper, we will focus on those WSNs whose beacons are designed to be immobile because the mobile beacon solution frequently requires additional localization and driving components on beacons, which may not be available for many low-cost, low-power WSN systems.

The "beacons are reliable" assumption frequently does not hold for many WSN systems deployed in unstable environments because beacons may be moved unexpectedly to other positions. In this occasion, the beacons will no longer be reliable and the localization result will be seriously affected. Srirangarajan et al. considered the effect of inaccurate beacon positions and proposed a distributed localization algorithm based on second-order cone programming (SOCP) relaxation [16]. The proposed algorithm tries to refine beacons' positions by using the relative distance information exchanged with their neighbors. Fan et al. presented a centralized localization algorithm to deal with both inaccurate TOA estimation and positioned beacons [17]. Vemula et al. developed a distributed localization mechanism that incorporates beacon position uncertainty using iterative and Monte Carlo sampling-based methods [18]. Lui et al. devised a semidefinite programming (SDP) algorithm for TOA-based localization in the presence of beacon positions and signal propagation speed uncertainties [19]. The authors also presented a simplified form of the devised algorithm based on the assumption that beacon position errors are independently and identically distributed. Feng et al. designed a robust multihop localization scheme based on trust evaluation to alleviate the effect of inaccurate beacon positions on localization accuracy [20]. In the proposed algorithm, sensor nodes use the trust degrees of beacons as weights in localization to achieve robust estimation of their positions. In the above works, researchers tried to optimize the performance of a specific localization method in case that beacons' positions are no longer reliable. As a comparison, we focus on the beacon drifting detection problem in this paper and try to investigate a distributed detection algorithm that can be applied to existing localization mechanisms to improve localization accuracy.

The most related work to ours is by Kuo et al. [21, 22], who proposed beacon movement detection (BMD) algorithm to detect the unexpected movements of beacons. The basic idea of BMD algorithm is to let the beacons record the variance of RSSI measurement results between each other and report to a calculation center to determine the moved beacons. Like all other centralized algorithms, BMD algorithm will bring a heavy communication burden in collecting detection information from beacons. At the same time, it is also difficult to ensure the completeness of the detection information collected along a long routing path. In order to alleviate the problem, we propose a distributed algorithm to determine the moved beacons in this paper.

3. Algorithm Design

We frequently use RSSI measurement value to estimate the distance between nodes. Typically, greater variance of RSSI measurement value over time indicates greater possibility of relative movement between nodes. Thus, we can judge if there is a beacon drifting event through monitoring the variance of RSSI measurement value between beacons. In this section, we will try to investigate a distributed beacon drifting detection algorithm, in which the beacon itself will determine if it has moved. This distributed algorithm does not require a calculation center and, thus, will not bring heavy communication burden in a large-scale network.
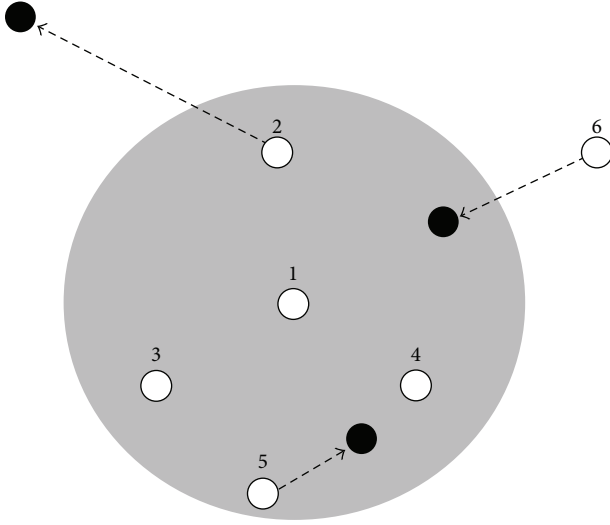
Figure 1: An example of fault detection.

Nevertheless, using RSSI measurement value to distributedly deduce the beacon drifting event may cause serious fault detection problem, which can be described as below:

(i) RSSI measurement value is not always trustable; it may vary even if there is no relative movement between two beacons.

(ii) Even if the variance of RSSI measurement value can precisely describe the relative movement between beacons, it is also difficult for us to determine exactly which one moves or both of them move. Figure 1 gives an example. In the figure, white dots indicate the original positions of beacons, and the black dots indicate the new positions of beacons after movement. The shadowed area indicates the communication range of beacon 1. We can see that no movement occurred on beacons 1, 3, and 4. Beacon 5 has moved, but is still within the communication range of beacon 1. Beacon 2 moved outside the communication range of beacon 1. Beacon 6 moved into the communication range of beacon 1 and became a new neighboring beacon of beacon 1. Even if the RSSI measurement is accurate (i.e., beacons can accurately detect that if there is a relative movement between each other), different beacons may have different conclusions on movement detection. For instance, the relative distances between beacon 1 and beacons 2, 5, and 6 have changed due to the movements of beacons 2, 5, and 6. For beacon 1, it will find that beacons 2, 5 and 6 have moved. But for beacons 2, 5, and 6, they may find that beacon 1 has moved, but not themselves. This conflicting conclusion will make it difficult for a distributed algorithm to find the correct set of moved beacons.

In order to alleviate the fault detection problem, we designed a cooperative algorithm for beacons including the self-scoring phase and negotiation phase. Generally, the algorithm works as follows:

(1) each beacon calculates an initial score for itself. This score indicates the movement possibility of the beacon based on the RSSI monitoring results between itself and neighboring beacons.

(2) Each beacon negotiates with neighboring beacons by telling them its score and adjusts its score based on the received neighboring beacons' scores. This score exchanging process can let beacons share their monitoring results and determine who have moved cooperatively.

(3) After several rounds of negotiation, each beacon gets a final score. The beacon will determine if itself has moved based on its final score.

3.1. Beacon Self-Scoring Algorithm. The beacon self-scoring algorithm is designed based on the monitoring results of RSSI measurement value variance over a period of time. The basic principle of the designed algorithm is for a certain beacon, if there is a large variance of the RSSI measurement value between itself and its neighboring beacons, its position information tends to be unreliable.

Given a network with N beacons, beacon b_i can record its RSSI measurement value between itself and its neighboring beacons in a RSSI vector $O_{i(t)}$ at time t, as shown in (1), in which M_i is the number of neighboring beacons of the beacon b_i, $RSSI_{ij(t)}$ is the RSSI measurement value between the beacons b_i and b_j at time t, s is RF receiver sensitivity of beacons, and $o_{ij(t)}$ is the jth element in the RSSI vector $O_{i(t)}$. $o_{ij(t)}$ indicates the monitoring result between the beacon b_i and beacon b_j at time t:

$$o_{ij(t)} = \begin{cases} RSSI_{ij(t)}, & \text{if } b_i \text{ can hear } b_j \\ s, & \text{otherwise} \end{cases} \quad (0 < i \le N, \ 0 < j \le M_i).$$

(1)

Beacons update their RSSI vectors periodically, and calculate changed RSSI vectors and unchanged RSSI vectors, as shown in (2), in which $P_{i(t)}$ and $Q_{i(t)}$ are the changed RSSI vector and unchanged RSSI vector of beacon b_i at time t, respectively. $p_{ij(t)}$ and $q_{ij(t)}$ are the jth elements of $P_{i(t)}$ and $Q_{i(t)}$, respectively. δ is a threshold:

$$p_{ij(t)} = \begin{cases} 1, & \text{if } \left| o_{ij(t)} - o_{ij(0)} \right| > \delta, \\ 0, & \text{otherwise}, \end{cases}$$

$$q_{ij(t)} = \begin{cases} 1, & \text{if } \left| o_{ij(t)} - o_{ij(0)} \right| \le \delta \ \left(o_{ij(t)} \ne s \text{ or } o_{ij(0)} \ne s \right) \\ 0, & \text{otherwise} \end{cases}$$

$$(0 < i \le N, \ 0 < j \le M_i).$$

(2)

We denote the number of "1" in $P_{i(t)}$ by $C_{Pi(t)}$, and the number of "1" in $Q_{i(t)}$ by $C_{Qi(t)}$.

Then, the score can be calculated according to (3), in which the notation $|\cdot|$ means calculating the absolute value of each element in the vector, $Sr_{i(t)}$ is the score of beacon b_i at time t, and k is the weight of unmoved neighboring beacons:

$$Sr_{i(t)} = \left| O_{i(0)} - O_{i(t)} \right| P_{i(t)}^T - kC_{Qi(t)} \quad (0 < i \le N).$$
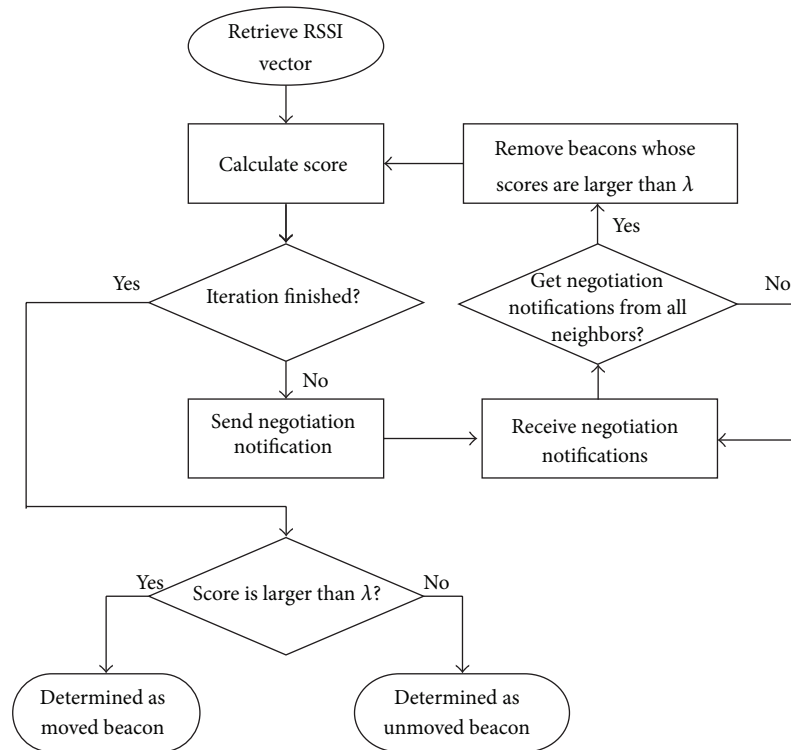
(3)

FIGURE 2: Beacon-to-beacon negotiation mechanism.

According to (3), for a certain beacon, if there is a larger change in RSSI measurement value, or the number of its neighboring beacons that are detected to have a large change in RSSI measurement value increases, the score will be higher. As a result, the score reflects the possibility of drifting from the beacon's point of view. A high score indicates that the beacon has a high possibility of drifting.

3.2. Beacon-to-Beacon Negotiation Mechanism.
The beacon self-scoring algorithm calculates the possibility of drifting by the beacon itself. Obviously, if there are more neighboring beacons are drifting, the self-scoring result is more unreliable. Thus, we designed a negotiation mechanism to alleviate the problem, which can be described as below:

(1) each beacon broadcasts a negotiation notification which contains its self-scoring result to its neighboring beacons.

(2) Each beacon that received negotiation notifications from all its neighboring beacons repeats the self-scoring process but excludes those beacons whose scores are larger than a threshold λ. Through this operation, a beacon can use the RSSI monitoring results between itself and those neighboring beacons which are more likely to be unmoved in calculating score, and thus the detection accuracy can be increased.

(3) The above two steps can be repeated for several times. If the iteration finished and the score of one beacon is still above λ, the beacon will determine itself as a moved beacon, or it will determine itself as an unmoved beacon.

Figure 2 describes the negotiation process. Higher iteration times may further increase detection accuracy, but will also bring a heavier communication and calculation burden. We will discuss the iteration times selection in Section 4.

3.3. Determining Algorithm Parameters.
The number of drifting beacons and their moving patterns (e.g., distance) might be different in different networks. Figure 3 shows the simulation results of RSSI measurement value variances in different networks, in which plus signs are the original positions of beacons, circles are the positions of beacons after movement, dotted lines are the directions of movements, and a solid line indicates that there is a relative movement detected between two beacons. It can be seen that when the number of moved beacons is small and the number of all beacons is large, moved beacons have much more connected solid lines than unmoved beacons, and thus we can easily distinguish these two sets of beacons. However, when the number of moved beacons and the number of all beacons are close, it will be difficult to distinguish moved and unmoved beacons. In the discussion below, we will discuss how to choose parameters in the proposed detection algorithm.

RSSI Variance Threshold (δ). Through adjusting the RSSI variance threshold δ, we aim to realize the following: (1) if the distance between beacons does not change, minimize the possibility of fault detection of beacon drifting; (2) if the distance between beacons does change, minimize the

(a) High beacon density, small number of drifting beacons

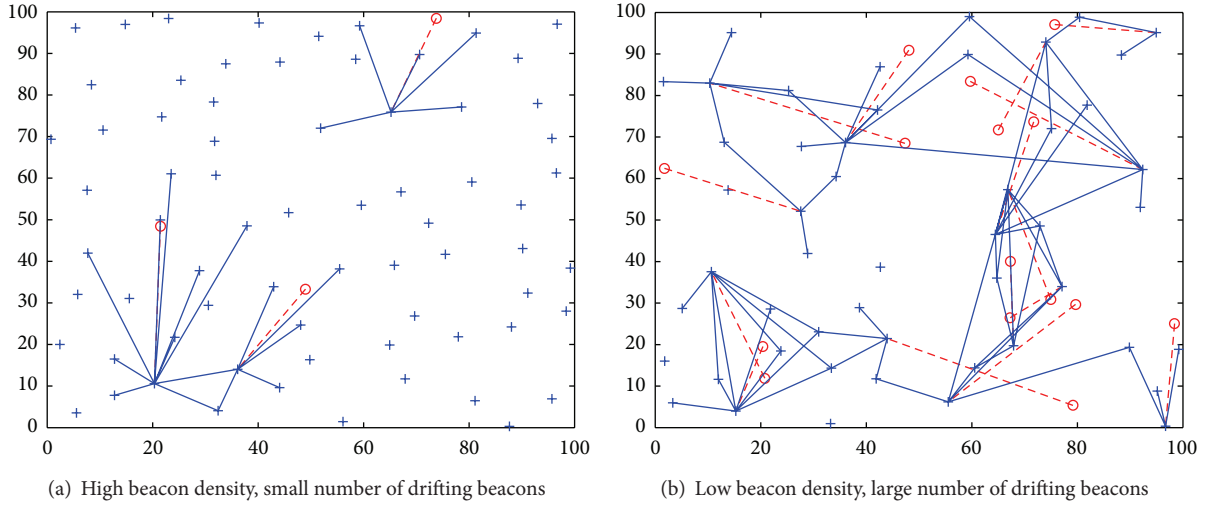(b) Low beacon density, large number of drifting beacons

Figure 3: The variances of RSSI measurement value in different networks.

possibility of failure to detect beacon drifting. According to previous research results, RSSI measurement value follows the distribution of $E \sim N(E_0 - 10n_p \lg(d/d_0), \sigma^2)$, in which E and E_0 are the RSSI measurement values in case that the distances between two beacons are respectively d and a short reference distance d_0, and n_p is the path loss exponent [23]. We chose the confidence interval as 95.4% and set the RSSI variance threshold $\delta = 2\sigma$. In this case, if there is no beacon drifting event, the possibility of RSSI measurement value variance caused by RSSI noise exceeding the threshold is only about 5%.

Weight of Unmoved Neighboring Beacons (k). Through adjusting the weight of unmoved neighboring beacons k, we aim to be able to keep the score around threshold λ if it is difficult to judge whether there is a beacon drifting event occurs. For example, if half the neighboring beacons are moved and the rest neighboring beacons are unmoved, it will be difficult to judge if one beacon is moved. Considering (3), when the number of moved beacons and unmoved beacons are equal, the score should be around the threshold λ. For instance, if we use 0 as the threshold λ, then k should be equal to the mathematical expectation of all elements of $|O_{i(0)} - O_{i(t)}|$. However, it is almost impossible to estimate the above mathematical expectation in every deployment. Because the change of RSSI vector over time largely depends on the beacons' receiver sensitivity, we can set k to the absolute value of αs in deployment, in which s is the RF receiver sensitivity. Our experimental exploration shows that 1/4 is an appropriate value for α.

4. Evaluation Results

In evaluation, we assume that N beacons are deployed randomly in a $L \times L$ square region, and N_d beacons are drifting randomly within circular areas with radii of D. The maximum communication distance between beacons is 20 m; λ is set to 0.

We use success rate R_s and error rate R_e to evaluate the detection performance. The calculation of R_s and R_e is given in (4), in which B_d is the set of actually moved beacons, B_{dg} is the set of moved beacons generated by the proposed algorithm, and B is the set of all beacons:

$$R_s = \frac{\text{card}\left(B_{dg} \cap B_d\right)}{\text{card}\left(B_d\right)},$$

$$R_e = \frac{\text{card}\left(B_{dg} \cap \left(B - B_d\right)\right)}{\text{card}\left(B_d\right)}. \tag{4}$$

In the discussion below, we will base our evaluation on simulation because it is frequently difficult to build an accurate analytical model for a complex networked system due to its traffic and topology dynamics [24].

4.1. Selection of the Iteration Times of Negotiation. In this subsection, we will study the impact of different iteration times of negotiation on the performance of the proposed algorithm. In simulation, L is set to 100 m, N is set to 50, and D is set to 40 m, and we tested the algorithm performance in case that N_d varies from 1 to 25. Figure 4 gives the test results.

From the figure, we can observe that the noniterative version of the proposed algorithm (i.e., no negotiation is executed in calculating score) gets a high success rate, and at the same time, a relatively high error rate. The success rate almost always keeps at 100%, but maximum error rate also reaches about 60%. This is because that the self-scoring algorithm actually tries to search for as many suspected moved beacons as possible, and thus one beacon has a high possibility of mistakenly judging itself as a moved beacon. We will then check the performance of the proposed algorithm with negotiation. We can see that if the negotiation process iterates once, we can get a much lower error rate (although at this time the success rate drops a little). When we increase the iteration times, it does not bring any obvious performance improvement, but causes a high communication burden. As
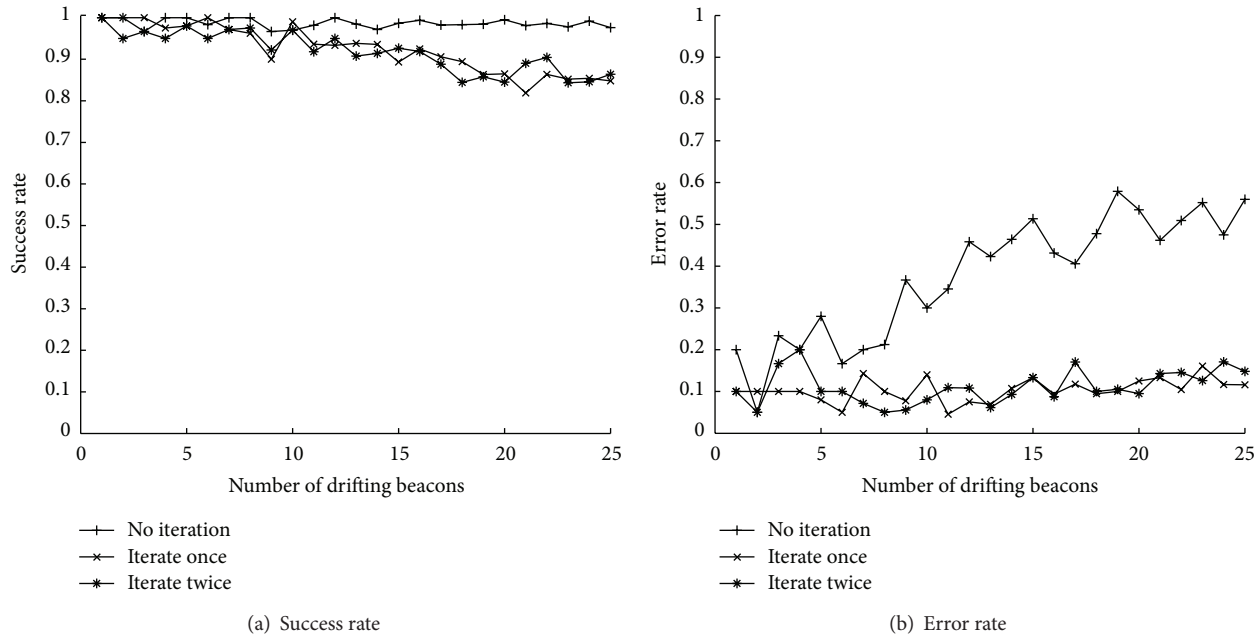
(a) Success rate

(b) Error rate

FIGURE 4: Impact of iteration times of negotiation on the proposed algorithm.
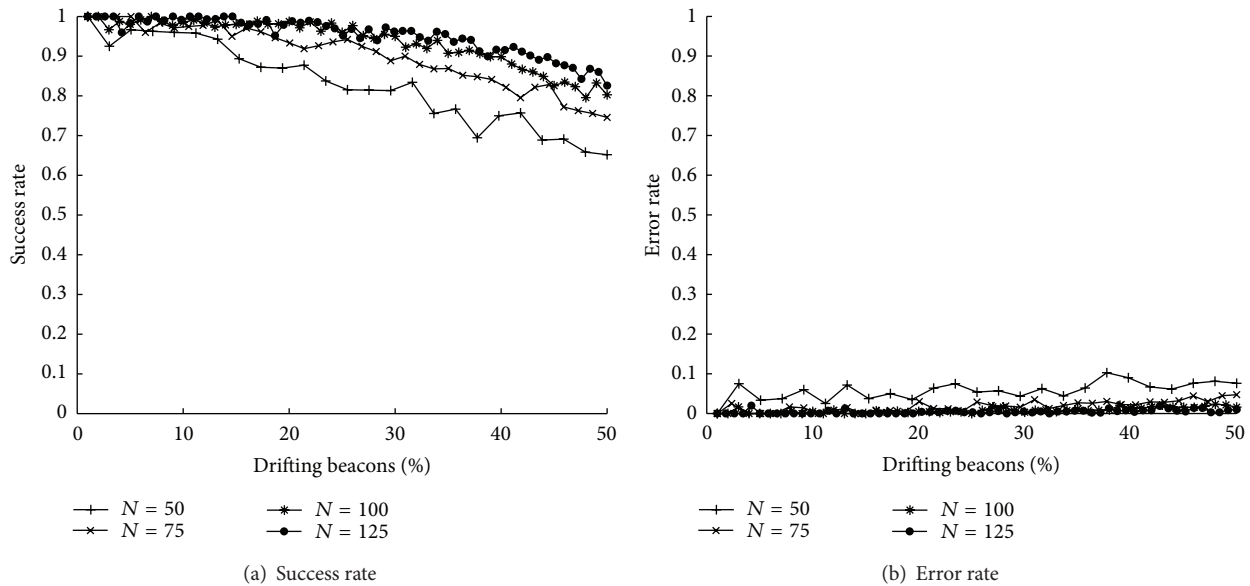

(a) Success rate

(b) Error rate

FIGURE 5: Performance of the proposed algorithm in different networking scenarios.

shown in the figure, when we iterate the negotiation process twice, the algorithm performs almost the same as that whose iteration times is set to 1.

As a result, we typically choose to execute the negotiation process once in detecting drifting beacons and will use this setting in later discussions.

4.2. Algorithm Performance in Different Networking Scenarios. In this subsection, we will study the performance of the proposed algorithm in different networking scenarios. The different networking scenarios we discuss here include (1)

different percentages of drifting beacons out of all beacons in the network; (2) different beacon densities (the number of beacons in a given size network coverage area). In the simulation, L is set to 100 m, D is set to 40 m, and we compare the algorithm performance in case that the percentage of drifting beacons varies from 1% to 50%, and N is respectively set to 50, 75, 100, and 125. Figure 5 shows the simulation results. It can be observed that if the percentage of drifting beacons is low, we can get an ideal detection success rate and error rate. With the increase of the percentage of drifting beacons, both the success rate and error rate degenerate. However, if there are no more than 30% beacons are drifting,

(a) Different number of beacons

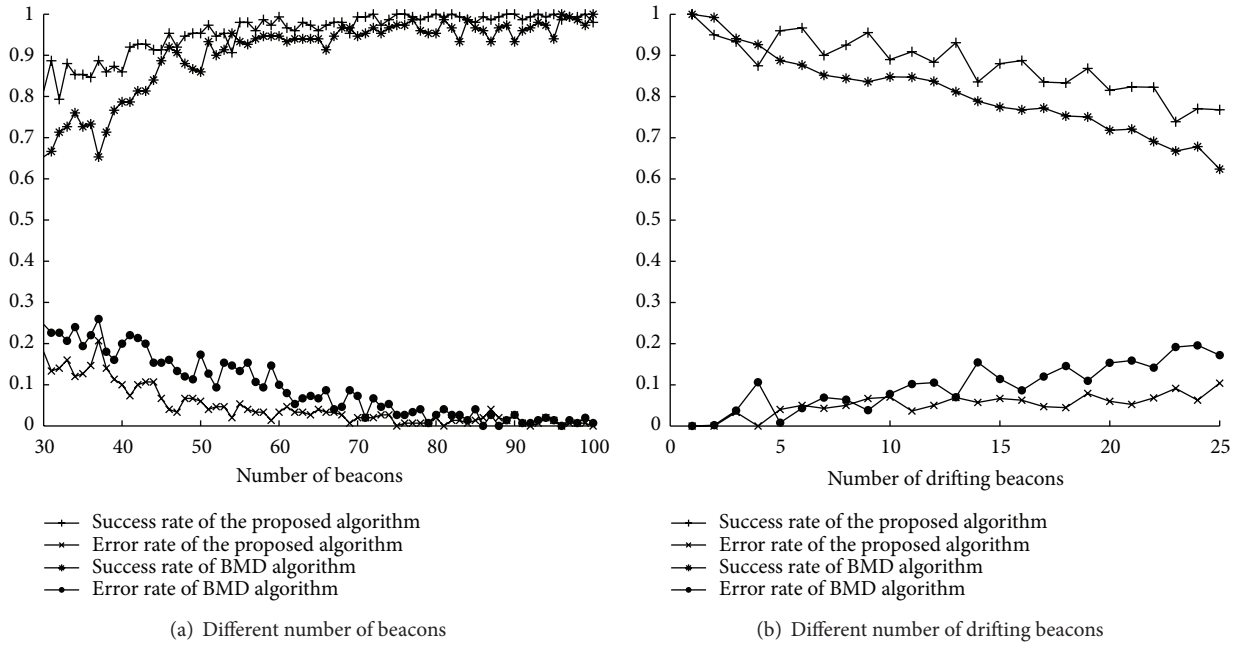(b) Different number of drifting beacons

Figure 6: Detection performance comparison.

the worst detection success rate is still higher than 0.8. At the same time, the worst detection error rate is always within 0.1 even if there are 50% beacons are drifting. The simulation results also show that the proposed algorithm performs better in high-density networks. For instance, when 50% of beacons are drifting, the success rate and error rate of the proposed algorithm are about 0.65 and 0.08 if there are 50 beacons in total in the network. As a comparison, if there are 125 beacons in the network, the success rate and error rate of the proposed algorithm can be improved to about 0.83 and 0.01, respectively.

4.3. Algorithm Comparison. In this subsection, we compare the detection performance and communication overhead between the proposed algorithm and BMD algorithm.

Figure 6 provides the detection performance comparison results, in which L is set to 100 m, and D is set to 40 m. In Figure 6(a), N_d is set to 15, and N varies from 30 to 100. In Figure 6(b), N is set to 50, and N_d varies from 1 to 25. It can be seen that the proposed algorithm performs better than BMD algorithm in both cases.

Figure 7 shows the communication overhead comparison results, in which L varies from 100 m to 200 m, N is proportional to L^2 to keep the beacon density constant, D is set to 40 m, and N_d is set to 5. Here, the communication overhead indicates the total number of delivered data packets in performing drifting beacon detection. It can be seen that the communication overhead of the proposed algorithm is almost the same as that of BMD algorithm when the network scale is small. With the increase of the network scale, the communication overhead of BMD algorithm increases much faster than that of the proposed algorithm. This is because
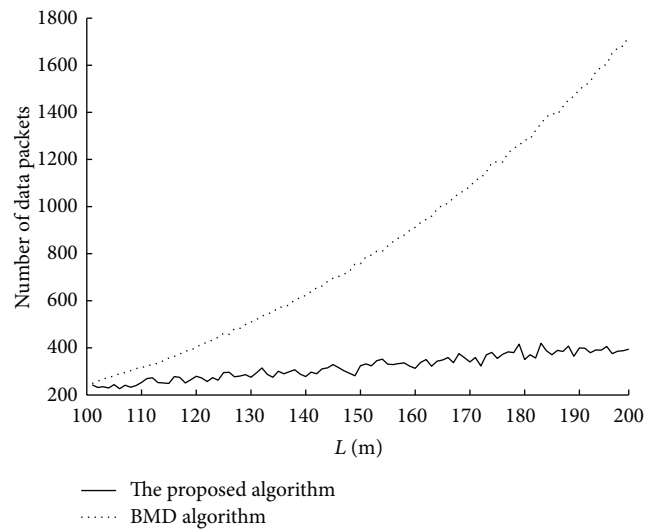

Figure 7: Communication overhead comparison.

BMD algorithm is a centralized algorithm, and beacons have to report monitored information to a calculation center to perform detection. As a result, when the network scale increases, the hop counts from beacons to the calculation center increase, and the number of delivered data packets increases fast. As a comparison, the proposed algorithm performs detection in a distributed manner, one beacon only needs to communicate with neighboring beacons to determine if it has moved, and thus the communication overhead increases much slower than centralized algorithms.

5. Concluding Remarks

Beacon drifting (the unexpected movement of beacon) greatly affects localization accuracy. In this paper, we proposed a distributed beacon drifting detection algorithm. The proposed algorithm can be applied to existing localization algorithms to avoid using moved beacons in calculating the positions of sensor nodes, and thus the localization accuracy can be improved. Both beacon self-scoring and beacon-to-beacon negotiation mechanisms were designed to achieve high detection accuracy, and at the same time, low operating overhead. Evaluation results show that the proposed algorithm works well in different networking scenarios and outperforms the existing centralized algorithm in both detection accuracy and communication overhead.

Acknowledgments

References

[1] 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.

[2] H. Shi, W. Wang, N. Kwok, and S. Chen, "Game theory for wireless sensor networks: a survey," *Sensors*, vol. 12, no. 7, pp. 9055–9097, 2012.

[3] J. Polastre, R. Szewczyk, A. Mainwaring, D. Culler, and J. Anderson, "Analysis of wireless sensor networks for habitat monitoring," *Wireless Sensor Networks*, pp. 399–423, 2004.

[4] P. Lu, S. Chen, and Y. Zheng, "Artificial intelligence in civil engineering," *Mathematical Problems in Engineering*, vol. 2012, Article ID 145974, 22 pages, 2012.

[5] Y. Zheng, S. Chen, Y. Lin, and W. Wang, "Bio-inspired optimization of sustainable energy systems: a review," *Mathematical Problems in Engineering*, vol. 2013, Article ID 354523, 12 pages, 2013.

[6] G. Mao, B. Fidan, and B. D. O. Anderson, "Wireless sensor network localization techniques," *Computer Networks*, vol. 51, no. 10, pp. 2529–2553, 2007.

[7] P. Bahl and V. Padmanabhan, "Radar: an in-building rf-based user location and tracking system," in *Proceedings of the IEEE 19th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '00)*, pp. 775–784, March 2000.

[8] A. Savvides, C. C. Han, and M. B. Strivastava, "Dynamic fine-grained localization in ad-hoc networks of sensors," in *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking (MOBICOM '01)*, pp. 166–179, July 2001.

[9] H. Shi, W. Wang, N. Kwok, and S. Chen, "Adaptive-indexed divisible load theory for wireless sensor network workload allocation," *International Journal of Distributed Sensor Networks*, vol. 2013, Article ID 484796, 18 pages, 2013.

[10] M. L. Sichitiu and V. Ramadurai, "Localization of wireless sensor networks with a mobile beacon," in *Proceedings of the IEEE International Conference on Mobile Ad-Hoc and Sensor Systems (MASS '04)*, pp. 174–183, October 2004.

[11] E. Goldoni, A. Savioli, M. Risi, and P. Gamba, "Experimental analysis of RSSI-based indoor localization with IEEE 802.15.4," in *Proceedings of the IEEE European Wireless Conference (EW '10)*, pp. 71–77, April 2010.

[12] L. Girod and D. Estrin, "Robust range estimation using acoustic and multimodal sensing," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '01)*, pp. 1312–1320, November 2001.

[13] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan, "The cricket location-support system," in *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MOBICOM '00)*, pp. 32–43, August 2000.

[14] D. Niculescu and B. Nath, "DV based positioning in ad hoc networks," *Telecommunication Systems*, vol. 22, no. 1–4, pp. 267–280, 2003.

[15] S. Li, D. Lowe, X. Kong, and R. Braun, "Wireless sensor network localization algorithm using dynamic path of mobile beacon," in *Proceedings of the 17th IEEE Asia-Pacific Conference on Communications (APCC '11)*, pp. 344–349, October 2011.

[16] S. Srirangarajan, A. Tewfik, and Z. Q. Luo, "Distributed sensor network localization using SOCP relaxation," *IEEE Transactions on Wireless Communications*, vol. 7, no. 12, pp. 4886–4895, 2008.

[17] R. Fan, H. Jiang, S. Wu, and N. Zhang, "Ranging error-tolerable localization in wireless sensor networks with inaccurately positioned anchor nodes," *Wireless Communications and Mobile Computing*, vol. 9, no. 5, pp. 705–717, 2009.

[18] M. Vemula, M. F. Bugallo, and P. M. Djurić, "Sensor self-localization with beacon position uncertainty," *Signal Processing*, vol. 89, no. 6, pp. 1144–1154, 2009.

[19] K. W. K. Lui, W. K. Ma, H. C. So, and F. K. W. Chan, "Semi-definite programming algorithms for sensor network node localization with uncertainties in anchor positions and/or propagation speed," *IEEE Transactions on Signal Processing*, vol. 57, no. 2, pp. 752–763, 2009.

[20] R. Feng, X. Guo, N. Yu, and J. Wan, "Robust multihop localization for wireless sensor networks with unreliable beacons," *International Journal of Distributed Sensor Networks*, vol. 2012, Article ID 972101, 13 pages, 2012.

[21] S. P. Kuo, H. J. Kuo, Y. C. Tseng, and Y. F. Lee, "Detecting movement of beacons in location-tracking wireless sensor networks," in *Proceedings of the IEEE 66th Vehicular Technology Conference (VTC-Fall '07)*, pp. 362–366, October 2007.

[22] S. P. Kuo, H. J. Kuo, and Y. C. Tseng, "The beacon movement detection problem in wireless sensor networks for localization applications," *IEEE Transactions on Mobile Computing*, vol. 8, no. 10, pp. 1326–1338, 2009.

[23] N. Patwari, J. N. Ash, S. Kyperountas, A. O. Hero, R. L. Moses, and N. S. Correal, "Locating the nodes: cooperative localization in wireless sensor networks," *IEEE Signal Processing Magazine*, vol. 22, no. 4, pp. 54–69, 2005.

[24] S. Chen, W. Huang, C. Cattani, and G. Altieri, "Traffic dynamics on complex networks: a survey," *Mathematical Problems in Engineering*, vol. 2012, Article ID 732698, 23 pages, 2012.