A Distributed Location Estimating Algorithm for Wireless Sensor Networks

Jang-Ping Sheu, Jian-Ming Li
National Central University
Department of CSIE
Chung-Li, 320 Taiwan
sheujp@csie.ncu.edu.tw

Chih-Shun Hsu
Nanya Institute of Technology
Department of CSIE
Chung-Li, 320 Taiwan
chison.hsu@msa.hinet.net

Abstract

The location estimation is a fundamental and essential issue for wireless sensor networks(WSNs). In this paper, we assume that only a few sensor nodes (named as beacon nodes) get their locations by Global Positioning System (GPS) and the remaining nodes without GPS (named as normal nodes) need to estimate their own locations by gathering the nearby neighboring information. Existing works are either too costly or not accurate enough. To improve previous works, we propose a distributed location estimation algorithm for WSNs. In our algorithm, each node without location information only needs to collect the location information of neighboring nodes and use simple computation to estimate its location. Besides, we improve the accuracy of the normal node's estimative region by discarding the communication area of the beacon node (named as the farther neighboring beacon node), which does not cover the normal node, from the original estimative region. We derive some rules to adjust the estimative region according to the relative location of the normal node and the farther neighboring beacon node. Simulation results show that the proposed algorithm achieves better accuracy of estimative locations.

1 Introduction

Many applications and communication protocols [12], [11], [14] of the wireless sensor networks(WSNs) are based on the location information of sensor nodes. Therefore, how to get the location knowledge of sensor nodes based on limited resources is an important issue for WSNs [7].

Many location estimation algorithms for WSNs have been proposed recently. The location estimation algorithms for WSNs can be categorized as range-based and range-free schemes. The range-based scheme determines the distance between two different sensor nodes based on a variety of information, such as Time of Arrival (TOA) [2], Time Difference of Arrival (TDOA) [11], Angle of Arrival (AOA) [10], or Received Signal Strength Indicator(RSSI) [6], [1]. After the distance has been determined, the location can be estimated according to the distance information. The estimation of the above time-of-flight technologies can be affected by multi-path and noise. Moreover, these schemes often need to equip with additional hardware. Consequently, range-based schemes are impractical solutions for a resource limited wireless sensor networks.

Because of the drawbacks of ranged-based schemes, many range-free solutions of the positioning system are presented [9], [10], [3], [5], [13], [8], [4]. Based on the concept of distance vector routing (DV-routing), the DV Based Positioning System is proposed in [9], [10]. The known-location nodes, called anchors, broadcast their position packets throughout the network. Like DV-routing, each other nodes after receiving the location packet will maintain a shortest hop count table. Based on this method, each anchor can convert the distance, in hops, to physical distance and broadcasts the calculating result to neighboring nodes. However, the DV-based scheme requires lots of communications. Additionally, it must work in a network which is dense enough.

Another range-free location estimation scheme proposed in [3], named as the Convex Position Estimation algorithm, uses the approximate location information. When an unknown-position node hears some anchors nearby, it indicates that the unknown-location node must be deployed within the overlapping area of these anchors' communication region. According to these anchors locations and communication ranges, the possible location of the unknown-location node can be estimated. The Convex Position Estimation algorithm needs a central controller to estimate the location of every unknown-location node and flood the location information to every sensor node. Although the algorithm can compute a global solution, it has a poor scalability due to the heavy communication cost.

A range-free localization algorithm named as APIT is proposed in [5]. The APIT scheme repeats the PIT (Point-

Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC'06)
0-7695-2553-9/06 $20.00 © 2006 IEEE

In-Triangulation) test with different audible anchor combinations until the required accuracy is achieved or all combinations are exhausted. After the PIT test, the center of gravity (COG) of the intersection of all the triangles is calculated to determine the estimated position. It is shown that the APIT scheme performs well when an irregular radio pattern and random node placement are considered. It can achieve acceptable localization errors with low communication overhead.

To improve centralized algorithms, a distributed location estimation algorithm is proposed in [13]. This algorithm has a better scalability, but it does not discard the communication area of the beacon node, which does not cover the normal node, from the original estimative region, and thus not improving the accuracy of the estimative region. In references [8], [4], mobile beacon nodes are dispatched to improve the accuracy of localization, however, the beacon nodes in our protocol are static.

To improve the accuracy of the previous works, we propose a Distributed Location Estimation (DLE) algorithm for WSNs. In the proposed algorithm, each normal node gathers the nearby beacon nodes' locations and then estimates its own location. Since the normal node is a low cost, small size, and power limited devices equipped with a low-level microprocessor, it can not perform high-level computations. Consequently, the computation is simplified by replacing complicated functions with simple operations. Besides, we adjust the normal node's estimative region by discarding the communication area of the beacon node (named as the farther neighboring beacon node), which does not cover the normal node, from the original estimative region. Thus, the accuracy of the estimative location can be improved. The estimative region is adjusted according to the relative location of the normal node and the farther neighboring beacon node. Simulation results show that the proposed algorithm can improve the accuracy of the estimative location. To show the feasibility of our algorithm, we also implement our algorithm on the network simulator of TinyOS platform [7], which is an efficient and modular embedded software platform for the Mote Processor/Radio module family.

The rest of the paper is organized as follows. In section 2, we introduce our Distributed Location Estimation algorithm. Simulation results are shown in section 3. Section 4 concludes this paper.

2 Distributed Location Estimation Algorithm

In this section, we describe our distributed location estimation (DLE) algorithm for WSNs. The WSN is assumed to have the following characteristics:

- There are N sensor nodes in the network.
- Every sensor nodes has a unique ID.
- Sensor nodes are deployed irregularly.
- There are M beacon nodes in the network, where $0 < M < N$. Each of the beacon node is equipped with a GPS and thus has the knowledge of its own location. The other nodes are normal nodes, which have no location information.
- The power level of beacon nodes can be changed to higher power level and thus enlarges the communication range of beacon nodes to $(1 + \sqrt{2})r$, where r is the transmission radius of normal nodes.

2.1 Overview

In the proposed algorithm, the normal nodes estimate their own locations according to local information. The problem is how to estimate the normal nodes' locations accurately according to neighboring nodes' locations. Since the beacon nodes are equipped with GPS, the beacon nodes can broadcast their own locations to neighboring normal nodes. After receiving location information from beacon nodes, the normal nodes can estimate their own locations according to neighboring beacon nodes' locations. When a normal node receives packets from several beacon nodes, the normal node is within the overlapping area of these beacon nodes' communication regions. For the convenience of the estimation, we define the Estimative Rectangle as the minimum rectangle that covers the overlapping area. To simplified the estimation, we assume that the four edges of the Estimative Rectangle are parallel to X-axis or Y-axis. The location of the normal node is assumed to be in the center of the Estimative Rectangle. The estimative region can be further reduced according to which nearby beacon nodes not covering the normal node and then discards their communication areas from the estimative region. In the following subsections, we first describe the three phrases of our algorithm, and then we show how to calculate the Estimative Rectangle by simplified computation. Finally, we demonstrate how to adjust the estimative region, so that the accuracy of the estimative location can be improved.

2.2 Three Phrases of our Algorithm

There are three phases in our distributed location estimation algorithm. The three phrases are described as follows:
Phase 1:

1. Every beacon node raises its power level so that its communication range is extended to $(1+\sqrt{2})r$, where r is the transmission radius of normal nodes.

2. Every beacon node gathers the ID and location information of neighboring beacon nodes by exchanging beacon frames.

Phase 2:

1. Every node (including the beacon and normal nodes) transmits its beacon frame by using the original power level.

2. Every normal node gathers the beacon frames and records all its one-hop neighbors' ID and all the neighboring beacon nodes' locations.

3. The neighboring beacon nodes will also provide some other beacon nodes' locations, which is collected in Phase 1. If the beacon node is not a one-hop neighbor of the normal node, the beacon node is regarded as the farther neighboring beacon node of the normal node.

Phase 3:

1. After collecting neighboring information, the normal node can calculate its own Estimative Rectangle(ER) according to the algorithm described in section 2.3.

2. If the farther neighboring beacon node's communication range covers the ER, we can improve the accuracy of the original ER according the rules described in section 2.4.

3. When a normal node has calculated its own ER through the above steps, it regards the center point of ER as its own location. Then it will broadcast its own estimative location to its one-hop neighbors.

4. If the normal node does not have any neighboring beacon node and has received the estimative locations of its neighboring normal nodes, it will regard these neighboring normal nodes as beacon nodes and use their locations to estimate its own location.

In phase 1, the beacon node needs to extend its communication range to $(1+\sqrt{2})r$ so that the normal node can gather its farther neighboring beacon nodes' information from its neighboring beacon nodes. For example, in Fig. 1, the normal node B can only receive packets directly from the neighboring beacon node A in phase 2. Since the communication region of the beacon node C overlaps with the ER of node B, node B needs to get the location of node C, so that it can reduce the size of its ER by discarding the overlapping region from its ER. Since node B cannot directly communicate with node C, node B can only get the location of node C from beacon node A, whose communication range is greater than that of node B. Therefore, node A must gather the locations of the other beacon nodes, which may cover the ER of node B. As shown in Fig. 1,

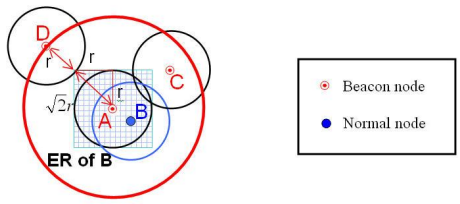

Figure 1. Extend communication range

node D is the farthest beacon node which may cover the ER of node B. If the extended communication range of node D is less than $(1+\sqrt{2})r$, node A will not be able to gather the location of node D. Therefore, each beacon node should extend its communication range to $(1+\sqrt{2})r$ during phase one, so that it can gather necessary information.

2.3 Estimative Rectangle (ER)

Since the sensor node is a low cost and power limited device equipped with a low-level microprocessor, it can not perform high-level computations. Therefore, the computation should be simplified, so that it can be executed on the sensor node. Since DLE is a distributed algorithm, we simplified the computation and only use basic arithmetic to calculate the Estimative Rectangle (ER). First, we simplify the calculation of the intersection points (the calculation is shown in section 2.3.1), and then we derive the four edges of the ER by comparing the coordinates of the intersection points and the upper, lower, left, and right most points of the circles (the algorithm is shown in section 2.3.2).

2.3.1 Calculate the Intersection Points

We assume that the communication range of the sensor node is a disk and the radius is r. We intend to solve the following problem by simple computations: There are two circles in the XY plane. The coordinates of the centers of the two circles are $O_1(x_1, y_1)$ and $O_2(x_2, y_2)$, respectively. The goal is to find the coordinates of the intersection points of the two circles.

The standard equation of a circle is $(x-h)^2+(y-k)^2=r^2$, where (x, y) is the coordinate of the point in the circle and (h, k) is the coordinate of the center of the circle. Therefore, the intersection points of two circles are the roots of the following equations:

$$(x-x_1)^2+(y-y_1)^2=r^2 \quad (1)$$

$$(x-x_2)^2+(y-y_2)^2=r^2 \quad (2)$$

To solve equations 1 and 2, we derive the equation of the line which passes through the intersection points of the two

circles. Subtract equation 1 with equation 2, we have the following equation:

$$(x-x_1)^2+(y-y_1)^2-((x-x_2)^2+(y-y_2)^2)=0 \quad (3)$$

Simplify the above equation we have:

$$2(x_2-x_1)x+2(y_2-y_1)y+x_1^2+y_1^2-x_2^2-y_2^2=0 \quad (4)$$

Equation (4) is the line which passes through the intersection points. We can simplify the original problem as finding the solution of equations 1 and 4. From equation 4, we can derive the following equation:

$$y=\frac{x_2-x_1}{y_1-y_2}x+\frac{1}{2}(y_1+y_2)+\frac{(x_1-x_2)(x_1+x_2)}{2(y_1-y_2)} \quad (5)$$

Let $a=\frac{x_2-x_1}{y_1-y_2}$ and $b=\frac{1}{2}(y_1+y_2)+\frac{(x_1-x_2)(x_1+x_2)}{2(y_1-y_2)}$, equation 5 can be simplified as $y=ax+b$. We replace y in equation 1 with $ax+b$ and derive the following equation:

$$(1+a^2)x^2+2(ab-ay_1-x_1)x+x_1^2+(b-y_1)^2-r^2=0 \quad (6)$$

Let $A=(1+a^2)$, $B=ab-ay_1-x_1$, and $C=x_1^2+(b-y_1)^2-r^2$, equation 6 can be simplified as $Ax^2+2Bx+C=0$ and the solution is $x=\frac{-B+\sqrt{B^2-AC}}{A}$ or $x=\frac{-B-\sqrt{B^2-AC}}{A}$. Replace y with $ax+b$, the coordinates of the two intersection points P and Q are $(\frac{-B+\sqrt{B^2-AC}}{A}, a\times\frac{-B+\sqrt{B^2-AC}}{A}+b)$ and $(\frac{-B-\sqrt{B^2-AC}}{A}, a\times\frac{-B-\sqrt{B^2-AC}}{A}+b)$, respectively.

2.3.2 Obtain the Estimative Rectangle(ER)

In this section, we intend to derive the four edges of the ER. First, we design an algorithm to derive the four edges of the ER of any two circles. After derived the ERs of every pair of circles, we can calculate the overlapping area of these ERs and derive the ER of multiple circles.

For any two given circles, we can use the solutions in section 2.3.1 to calculate coordinates of the two intersection points P and Q, and then we can derive the four edges of the ER of the two circles by comparing the coordinates of the intersection points and the upper, lower, left, and right most points of the circles. For the ease of describing our algorithm, we use ER_up, ER_right, ER_down, and ER_left to represent the four edges of the ER. Assume that r is the communication range, points O_1 and O_2 are centers of circles and their coordinates are (x_1, y_1) and (x_2, y_2), respectively. We define the following notations:

- **If** $(y_1 < y_2)$ **then** $down\_O = O_1$, $up\_O = O_2$ **else** $down\_O = O_2$, $up\_O = O_1$.
- **If** $(x_1 < x_2)$ **then** $left\_O = O_1$, $right\_O = O_2$ **else** $left\_O = O_2$, $right\_O = O_1$.

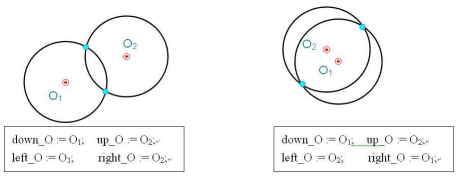

Figure 2. Examples of the notations

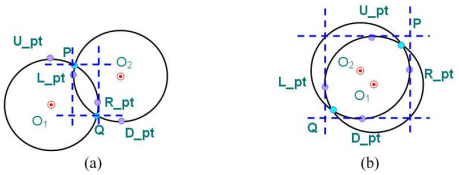

Figure 3. The examples of deriving ERs.

- $down\_x$: the X-axis coordinate of $down\_O$.
- up_x: the X-axis coordinate of up_O.
- $left\_y$: the Y-axis coordinate of $left\_O$.
- $right\_y$: the Y-axis coordinate of $right\_O$.

Fig. 2 shows the examples for the above notations. With the above notations, the coordinates of the upper, lower, left, and right most points of the circles can be derived as follows:

- $U\_pt = (down\_x, min(y_1, y_2) + r)$: the upper most point of the lower circle.
- $D\_pt = (up\_x, max(y_1, y_2) - r)$: the lower most point of the upper circle.
- $R\_pt = (min(x_1, x_2) + r, left\_y)$: the right most point of the left circle.
- $L\_pt = (max(x_1, x_2) - r, right\_y)$: the left most point of the right circle.

The upper, lower, left, and right most points of the circles in Fig. 2 are shown in Fig. 3. We can derive the four edges of the ER of the two circles by checking if U_pt, D_pt, R_pt, or L_pt belongs to the intersection region of the two circles or not. If U_pt belongs to the intersection region of the two circles, ER_up will pass through U_pt. Otherwise, ER_up will pass through the upper intersection points of the two circles. Similarly, if D_pt, R_pt, or L_pt belongs to the intersection region of the two circles, ER_down, ER_right, or ER_left will pass through D_pt, R_pt, or L_pt, respectively. By calculating the distance between U_pt and up_O, we can check if U_pt belongs to the intersection region or not. Similarly, we can calculate the distance between D_pt and $down\_O$, R_pt and $left\_O$, or L_pt and $right\_O$ to

check if D_pt, R_pt, or L_pt belongs to the intersection region or not.

For example, in Fig. 3(a), since the distances between R_pt and $left\_O$, and L_pt and $right\_O$ are all smaller than r, R_pt and L_pt belong to the intersection region of circles O_1 and O_2. Therefore, ER_right and ER_left will pass through R_pt and L_pt, respectively. However, the distances between U_pt and $down\_O$, and D_pt and up_O are all greater than r, U_pt and D_pt do not belong to the intersection region of circles O_1 and O_2. Therefore, ER_up and ER_down will pass through intersection points P and Q, respectively. Similarly, in Fig. 3(b), since U_pt, D_pt, R_pt, and L_pt all belong to the intersection region of circles O_1 and O_2, ER_up, ER_down, ER_right, and ER_left will pass through U_pt, D_pt, R_pt, and L_pt, respectively. The algorithm to derive the ER of the normal node covered by beacon nodes O_1 and O_2 is shown as follows:

Algorithm: $Estimative\_Rectangle(O_1, O_2)$
$d(a, b)$ is the distance between points a and b.
Begin

Calculate the intersection points $P(x_p, y_p)$ and $Q(x_q, y_q)$ of circles O_1 and O_2.

If $(d(U\_pt, up\_O) < r)$ **then** $ER\_up = min(y_1, y_2) + r$ **else** $ER\_up = max(y_p, y_q)$

If $(d(D\_pt, down\_O) < r)$ **then** $ER\_down = max(y_1, y_2) - r$ **else** $ER\_down = min(y_p, y_q)$

If $(d(R\_pt, right\_O) < r)$ **then** $ER\_right = min(x_1, x_2) + r$ **else** $ER\_right = max(x_p, x_q)$

If $(d(L\_pt, left\_O) < r)$ **then** $ER\_left = max(x_1, x_2) - r$ **else** $ER\_left = min(x_p, x_q)$

End

When a normal sensor node derives every ER of every pair of its neighboring beacon nodes, the intersection of these ERs is exactly the ER of the overlapping area of all these neighboring beacon nodes' communication regions. Assume that there are n beacon nodes cover the normal node. Therefore, there are total $C(n, 2) = n(n-1)/2$ pairs of circles overlap with each other. The coordinates of the upper left and lower right corners of the ith ER are (x_{ui}, y_{ui}) and (x_{vi}, y_{vi}), respectively. The coordinates of the upper left and lower right corners of the intersection region are (x_u, y_u) and (x_v, y_v), respectively. We have $x_u = \max(x_{ui})$, $y_u = \min(y_{ui})$, $x_v = \min(x_{vi})$, and $y_v = \max(y_{vi})$, where $i = 1 \ldots n(n-1)/2$. For example, in Fig. 4, there are three beacon nodes cover the normal node and there are total $C(3, 2) = 3$ pairs of circles overlap with each other. Assume that the coordinates of the upper left and lower right corners of the intersection region ER are (x_u, y_u) and (x_v, y_v), respectively. We have $x_u = \max(x_{u1}, x_{u2}, x_{u3}) = x_{u3}$, $y_u = \min(y_{u1}, y_{u2}, y_{u3}) = y_{u3}$, $x_v = \min(x_{v1}, x_{v2}, x_{v3}) = x_{v2}$, and $y_v = \max(y_{v1}, y_{v2}, y_{v3}) = y_{v1}$.

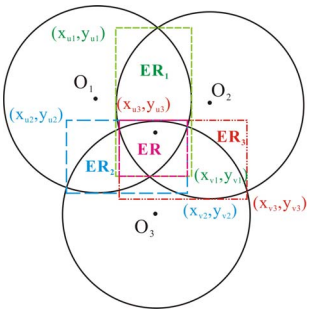

Figure 4. An example of deriving an ER from three overlapping ERs

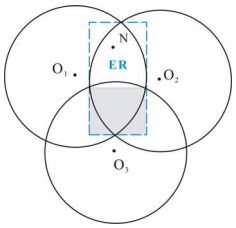

Figure 5. An example of a farther neighboring beacon node covers the ER of a normal node

2.4 Farther Improve the Accuracy of the ER

As mentioned in section 2.2, the normal nodes also gather the information of farther neighboring beacon nodes in phase 2. Every normal node can not directly communicate with any of its farther neighboring beacon nodes. If there are some farther neighboring beacon nodes' communication region cover a normal node's ER, it indicates that the normal node is not deployed in the overlapping area of its ER and the communication region of its farther neighboring beacon nodes. Therefore, the overlapping area of the ER and the farther neighboring beacon nodes' communication region can be discarded and thus improves the accuracy of the ER. Fig. 5 shows an example of a farther neighboring beacon node O_3, which covers the ER of a normal node N. The shaded region is O_3's communication region that can be discarded from the original ER. After the adjustment, the accuracy of the ER can be farther improved.

In the following paragraphs, we will first show how to adjust the ER of a normal node with only one farther neighboring beacon node, and then we will show how to adjust

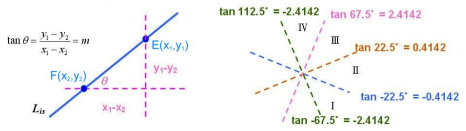

Figure 6. Calculations of the slope and 4 cases for different slopes.

the ER of a normal node with multiple farther neighboring beacon nodes. When there is only one farther neighboring beacon node, we need to derive the equation of the line (denoted as L_{is}) through the intersection points (denoted as P and Q) of the edge of the farther neighboring beacon node's communication region and the ER, so that we can discard the proper part of ER according to the slope of L_{is} and the relative location of the farther neighboring beacon node and the ER. If the ER's edge is adjusted to one of the intersection points, we may discard too much (or too few) area from the original ER. Therefore, we improve the accuracy of ER by adjusting the ER's edge (or corner) to the mid point of the line $\overline{PQ}$.

Assume that the slope of L_{is} is m and the angle between X-axis and L_{is} is θ. As shown in Fig. 6, the slope of L_{is} equals to the tangent value of θ. Several rules is derived to reduce the size of ER. For the ease of describing the rules, we classify the line L_{is} into 4 groups according to its slope. We have observed that when the line L_{is} is near vertical or horizontal (belongs to group 2 and 4), we need to adjust a vertical (or horizontal) edge of the ER to achieve a more accurate result. We move a vertical (when L_{is} belongs to group 4) or horizontal (when L_{is} belongs to group 2) edge of the ER to the mid point of the line $\overline{PQ}$, so that we will not discard too much (or too few) area from the original ER. On the other hand, when the slope of line L_{is} is closer 1 or -1 (belongs to group 1 and 3), adjusting either a vertical or a horizontal edge of the ER can not achieve a satisfactory result. Therefore, we need to adjust both a vertical and a horizontal edges of the ER to achieve a better result. We move a corner of the ER to the mid point of the line $\overline{PQ}$, so that we can achieve a more accurate result. If L_{is} belongs to group 1, we will adjust the upper right or lower left corner of the ER. Similarly, if L_{is} belongs to group 3, we will adjust the upper left or lower right corner of the ER. Which edge (or corner) of the ER need to be adjusted is chosen according to the relative location of the farther neighboring beacon node and the ER.

As Fig. 7 shows, if the slope is between -2.4142 and -0.4142, -0.4142 and 0.4142, 0.4142 and 2.4142, or ± 2.4142 and $\pm\infty$, the line L_{is} is classified to groups 1, 2, 3, or 4, respectively. According to the group of L_{is} and the relative location of the farther neighboring beacon node and the ER, there are 8 cases to reduce the size of the ER. As shown in

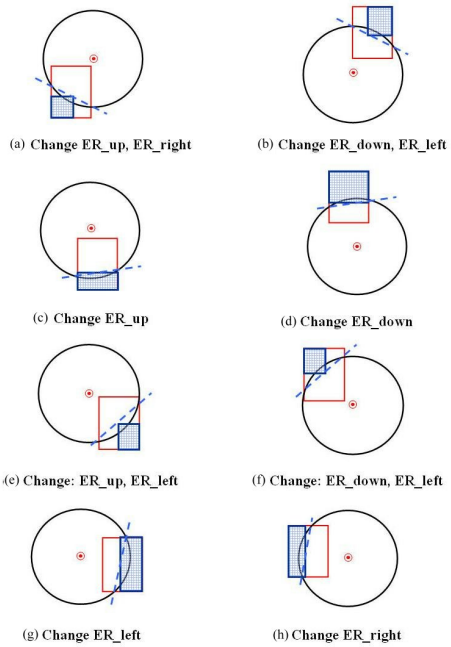

Figure 7. Rules to reduce the size of ER

Fig. 7(a), if the line L_{is} belongs to group 1 (θ is close to -45 degree) and the farther neighboring beacon node covers the upper right part of the ER, we need to adjust ER_up and ER_right. The upper right corner of the ER is moved to the mid point of the line $\overline{PQ}$. Similarly, if the line L_{is} belongs to group 1 and the farther neighboring beacon node covers the lower left part of the ER, we need to adjust ER_down and ER_left. The lower left corner of the ER is moved to the mid point of the line $\overline{PQ}$, as shown in Fig. 7(b). If the line L_{is} belongs to group 3 (θ is close to 45 degree), we can use the similar rules to reduce the size of ER, as shown in Figures 7(e) and (f). On the other hand, if the line L_{is} belongs to group 2 (θ is close to 0 degree) and the farther neighboring beacon node covers the upper part of the ER, we need to adjust ER_up, so that ER_up will pass through the mid point of the line $\overline{PQ}$, as shown in Fig. 7(c). Similarly, if the line L_{is} belongs to group 2 and the farther neighboring beacon node covers the lower part of the ER, we need to adjust ER_down, so that ER_down will pass through the mid point of the line $\overline{PQ}$, as shown in Fig. 7(d). If the line L_{is} belongs to group 4 (θ is close to 90 degree), we can use the similar rules to reduce the size of ER, as shown in Figures 7(g) and (h).

When there are multiple farther neighboring beacon nodes cover the ER of a normal node, we first sort these farther neighboring beacon nodes according to their slopes, and then the communication regions of the farther neighboring beacon nodes are discarded from the original ER one at a time according to the farther neighboring beacon

node's order. Each of the farther neighboring beacon node's communication region is discarded from the original ER according to the rules in previous paragraph. After discarding every farther neighboring beacon node's communication region from the original ER, we get the final ER. After deriving the ER, the normal node will regard the center of the ER as its estimative location.

3 Simulation Results

In order to demonstrate the efficiency of our Distributed Location Estimation algorithm, we compare the proposed DLE algorithm with the Convex Position Estimation (CPE) algorithm [3], which is a centralized algorithm.

We have developed a simulator using C in Linux system. The wireless sensor network in our simulation is a square region of side length $10r$, where r is the communication range of the normal nodes. The sensor nodes are placed randomly and each node is assigned with a unique ID. The simulation does not consider the influence of terrains and weathers. Additionally, we assume that the collision problem can be solved by the MAC layer protocol.

In our simulation, we compare the accuracy of the estimative location. To show the accuracy, we compute the mean error between the estimative and actual locations. The mean error of the estimative and the actual location can be calculated by the following formula:
$mean\_error = \frac{1}{n-m}\sum_{r=m+1}^{n}\sqrt{(x_e^k - x_r^k)^2 + (y_e^k - y_r^k)^2}$, where n is the number of sensor nodes, m is the number of beacon nodes, k is the ID of the normal node, (x_r, y_r) and (x_e, y_e) are the actual and the estimative coordinates of node k, respectively.

Two parameters are tunable in our simulation: the ratio of beacon nodes and the density of sensor nodes. Each result is obtained from the average of 50 simulation runs.

3.1 Impact of the Ratio of Beacon Nodes

In this section, we will show the impact of the ratio of beacon nodes on the mean error. We fix the number of beacon nodes and tune the number of beacon nodes. The area size of the wireless sensor network is set as $10r \times 10r$ and the total number of sensor nodes is set as 200. The number of beacon nodes is tuned between $10 \sim 95$. Fig. 8 shows the impact of the ratio of beacon nodes on the mean error. When the number of beacon nodes is smaller than 60, the mean error decreases as the number of beacon node increases. More beacon nodes can reduce the size of the normal node's ER and thus decreases the mean error. Our algorithm performs better than the CPE algorithm because we discards the area covered by farther neighboring beacon nodes from the ER and thus increases the accuracy of our algorithm. When the number of beacon nodes is greater than 60, the size of

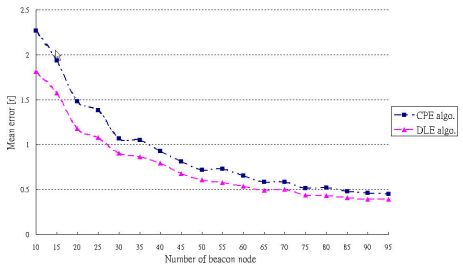

Figure 8. Ratio of beacon nodes vs. mean error

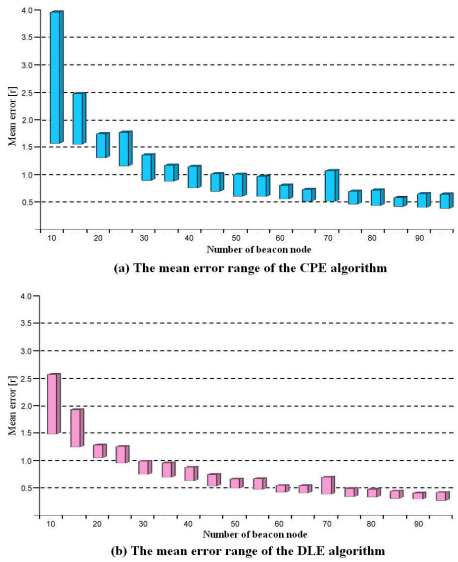

Figure 9. The experiment result of mean error range

ER can not be greatly reduced and thus the impact becomes smaller. Besides, the impact of farther neighboring beacon nodes also becomes smaller and thus the mean error of the two algorithms become close to each other.

To show the stability of our algorithm, we demonstrate the mean error value range of the CPE and DLE algorithms. As shown in Figures 9, the mean error range of our algorithm is smaller than that of the CPE algorithm, which indicates that our algorithm is more stable than the CPE algorithm.

3.2 Impact of the Density of Sensor Nodes

According to the results in section 3.1, when the ratio of beacon nodes is 30%, the efficiency is the best. Therefore, in this section, we fix the ratio of beacon nodes to 30% and the area size of the network to $10r \times 10r$. We

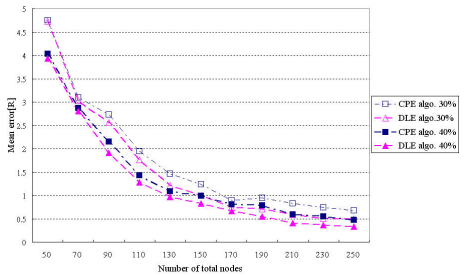

Figure 10. Node density vs. mean error

tune the total number of sensor nodes between 50 $\sim$ 250 and see its impact on the mean error. For comparison, we also make similar experiments with 40% of beacon nodes. Fig. 10 shows the impact of node density on the mean error. When the total number of sensor node is smaller than 150, both of the algorithm do not have good performance. When the node density is low (less than 150 nodes), each normal node has fewer neighboring beacon nodes, and thus do not have enough bounds to find accurate ER. However, when the node density becomes higher (more than 150 nodes), our algorithm performs better than the CPE algorithm in terms of mean error. Our algorithm can use fewer beacon nodes to achieve the same mean error as that of the CPE algorithm.

4 Conclusions

In the wireless sensor networks, the sensor node's location is a very important information for geographic-based applications, such as location tracking, geocast, location-based query, and sensor coverage problem. Most of the existing location estimation algorithm are either too costly or not accurate enough. To reduce the communication and computation cost and improve the accuracy, we have proposed a distributed location estimation algorithm for wireless sensor networks with a few beacon nodes. To implement our algorithm on real sensor nodes, we have simplified the computations of our algorithm and simulated it on the TOSSIM, which is a network simulator for TinyOS platform. Besides, we do not only use neighboring beacon nodes' locations to estimate the normal node's Estimative Rectangle, but also use farther neighboring beacon nodes locations to improve the accuracy of the Estimative Rectangle. Therefore, our algorithm can estimate more accurate locations for normal nodes. Simulation results have shown that our distributed algorithm can estimate locations more accurately than the CPE algorithm. When the number of sensor nodes is fixed and the number of beacon nodes increases, the mean error becomes smaller. When the ratio of beacon nodes is fixed and the number of sensor nodes increases, the mean error also becomes smaller.

References

[1] P. Bahl and V. N. Padmanabhan. Radar: An in-building rf-based user location and tracking system. *Proceedings of 19th Annual Joint Conference of the IEEE Computer and Communications Societies*, 2:775–784, March 2000.

[2] S. Capkun, M. Hamdi, and J.-P. HubauxInstitute. Gps-free positioning in mobile ad-hoc networks. *Proceedings of the 34th Annual Hawaii International Conference on System Science*, pages 3481–3490, January 2001.

[3] L. Doherty, K. S. J. Pister, and L. E. Ghaoui. Convex position estimation in wireless sensor networks. *Proceedings of 20th Annual Joint Conference of the IEEE Computer and Communications Societies*, 3:1655–1663, April 2001.

[4] A. Galstyan, B. Krishnamachari, S. Pattem, and K. Lerman. Distributed online localization in sensor networks using a moving target. *Proceedings of IEEE/ACM 3rd International Symposium on Information Processing in Sensor Networks*, pages 61–70, April 2004.

[5] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. abdelzher. Range-free localization schemes for large scale sensor networks. *Proceedings of 9th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, pages 81–95, September 2003.

[6] J. Hightower, R. Want, and G. Borriello. *SpotON: An Indoor 3D Location Sensing Technology Based on RF Signal Strength*. University of Washington, Department of Computer Science and Engineering, Seattle WA, February 2000.

[7] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for networked sensors. *Proceedings of the 9th International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 93–104, November 2000.

[8] L.-X. Hu and D. Evans. Localization for mobile sensor networks. *Proceedings of the 10th annual international conference on Mobile computing and networking*, Sepember 2004.

[9] D. Niculescu and B. Nath. Ad hoc positioning system (aps). *Proceeding of IEEE Global Telecommunications Conference*, 1:2926–2931, November 2001.

[10] D. Niculescu and B. Nath. Dv based positioning in ad hoc networks. *Journal of Telecommunication Systems*, 22:267–280, January-April 2003.

[11] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. *Proceedings of 6th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, pages 32–43, August 2000.

[12] S. Ray, R. Ungrangsi, F. D. Pellegrini, A. Trachtenberg, and D. Starobinski. Robust location detection in emergency sensor networks. *Proceedings of 22nd Annual Joint Conference of the IEEE Computer and Communications Societies*, 2:1044–1053, March-April 2003.

[13] Y. Shang and W. Ruml. Improved mds-based localization. *Proceedings of 23rd Annual Joint Conference of the IEEE Computer and Communications Societies*, March 2004.

[14] Y. Zou and K. Chakrabarty. "sensor deployment and target localization based on virtual forces". *Proceedings of 22nd Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 1293–1303, March-April 2003.