Localization in Wireless Sensor Networks Using Tabu Search and Simulated Annealing

S.Kazem Shekofteh
Department of Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Irankazem.
kazem.shekofteh@stu-mail.um.ac.ir

Maryam Baradaran Khalkhali
Department of Computer Engineering, Islamic Azad University, Mashhad Branch, Mashhad, Iran
maryam.baradaran.khl@gmail.com

Mohammad Hossien Yaghmaee
Department of Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Iran
yaghmaee@ieee.org

Hossein Deldari
Department of Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Iran
hd@ferdowsi.um.ac.ir

***Abstract*—Sensor localization is a fundamental and crucial issue for wireless sensor networks operation and management. Accurate self-localization capability is highly desirable in wireless sensor network. A fundamental problem in distance-based sensor network localization is whether a given sensor network is uniquely localizable or not. Flip ambiguity is a main problem that can make the sensor network not uniquely localized. It causes large errors in the location estimates. This paper proposes a method in which the localization is done through two steps. During the first step, tabu search (TS) is used to obtain an accurate estimation of the nodes' location. During the second step, simulated annealing algorithm (SAA) is used to refine the location estimates of those nodes that are likely to have flip ambiguity problem. The simulation results confirm that the proposed algorithm has better performance than that of the existing algorithms.**

Keywords- ***wireless sensor network, localization, tabu search, simulated annealing***

I. INTRODUCTION

The localization of sensor nodes in wireless sensor network (WSN) is an active research area for the past few years. Localization of sensor nodes is highly essential in WSN, especially in applications oriented towards the monitoring and tracking of physical phenomena [1]. In a WSN environment consisting of *m* sensor nodes, the objective of localization is to estimate the coordinates of n sensor nodes called target nodes using a priori information about the coordinates of *m-n* anchor or reference nodes. Such a cooperative localization system developed for WSN should be low cost, energy efficient, and of better accuracy.

The position estimation of the sensor nodes can be done either by using a geometrical approach that gives exact solutions to a set of simultaneous non-linear equations [2] or by using an optimization approach that minimizes the error in locating the coordinates of the target nodes [3, 4].

This paper proposes a localization method consisting of two steps for wireless sensor network that makes use of TOA based ranging and a global optimization technique based on Tabu Search (TS) for position estimation at the first step and uses Simulated Annealing (SA) optimization method at the second step to compensate the estimation error caused to some nodes because of flip ambiguity.

Our major contribution in this paper is the development of a novel, accurate and computationally efficient algorithm for WSN localization. It is found that the TS based localization is computationally efficient than the Simulated Annealing based localization, which is very significant for an energy constrained environment like WSN. Hence the TS method is selected to be used in the first step of the algorithm.

The rest of the paper is organized as follows: Section II provides a description of the related works; Section III gives a brief introduction to the TS and SA methods. The TOA-based ranging and TS-based position estimation (first step) and SA-based position estimation error correction (second step) are explained in Section IV. Simulation results are given in Section V. Conclusion is given in Section VI.

II. RELATED WORKS

A detailed survey on sensor networks is available in [1]. A review of the range measurement techniques and algorithms based on these measurements for WSN localization are provided in [5]. Ian Oppermann and Kegan Yu [2] discuss localization using UWB, based on TOA measurements for wireless embedded networks which employs direct calculation of target coordinates by solving hyperbolic equations and non linear optimization techniques for position estimation. These papers focus on Gauss Newton and Quasi-Newton methods for the minimization of an objective function which is a measure of the overall localization error. Kannan et al. Reference [4] propose a simulated annealing based WSN localization algorithm. In this paper the measured distance between neighboring nodes which is used in the cost function is blurred by introducing Gaussian noise into the true distance measurement. The magnitude of the additionally introduced noise is controlled by a constant noise factor.

978-1-4244-5586-7/10/$26.00 © 2010 IEEE

Savarese [6] proposed a method which has two phases, HOP-TERRAIN and refinement. The HOP-TERRAIN phase is allowing all nodes to arrive at initial location estimates. The refinement phase is an iterative algorithm. It uses the results of the HOPTERRAIN phase and the distance measurements of the immediate neighbors to do the least square trilateration.

Savvides [7] extended the single hop technique of GPS to multi-hop operation as in Niculescu's thus waiving the line of sight requirement with anchors. Non-anchor nodes collaborate and share information with one another over multiple hops to collectively estimate their locations. To prevent error accumulation in the network, they used least squares estimation with a Kalman filter to estimate locations of all non-anchor nodes simultaneously. To avoid converging at local minima, they used a geometrical relationship to obtain an initial estimate that is close to the final estimate. His algorithm is based on the assumptions that the distance measurements between the nodes and their neighbors are accurate.

Doherty [8] approached the problem using convex optimization based on semi-definite programming. The connectivity of the network was represented as a set of convex localizing constraints for the optimization problem. Pratik [3] extended this technique by taking the nonconvex inequality constraints and relaxed the problem to a semi-definite program. Tzu-Chen Liang improved Pratik's method further by using a gradient-based local search method [9]. All these semi-definite programming methods require rigorous centralized computation.

In this paper, we propose a two-phase algorithm which in the first phase, tabu search is used to obtain an accurate estimate of location. Then a second phase of optimization using simulated annealing, is performed only on those nodes that are likely to have flip ambiguity problem causing large localization errors.

III. FUNDAMENTALS OF THE ALGORITHM

A. Introduction to tabu search method

The Tabu Search [10, 11] a metaheuristic method for combinatorial optimization, proposed by Fred Glover in 1986. It is a dynamic neighborhood search technique that makes use of memory to drive the search by escaping from local optima and avoiding the cycling. The principal characteristic of the TS lies in its systematic use of flexible and adaptive memory which keeps track of the information of the previously visited solutions; whereas most of the commonly used search techniques keep only the value of the objective function corresponding to the best solution visited so far.

Any optimization problem for finding the minimum value of a real valued function can be formulated as Equation (1):

$$\min f(S),\ S \in X \qquad (1)$$

Where f denotes the objective function, S is a feasible solution and X is the set of entire feasible solutions of the problem. A neighborhood of a solution S is a set $N(S) \subseteq X$, where each solution $S' \in N(S)$ is reached from S by an operation called a move.

A solution S^* is a local optimum with respect to a given neighborhood N if $f(S^*) < f(S)$, $\forall s \in N(S^*)$. Local search methods like gradient search explore the neighborhood for improving solutions until a local optimum is found. The main disadvantage of this method is that it may stop exploration at any local minimum.

The TS method addresses this problem and improves the efficiency of the exploration by keeping track of the history of the search in addition to the local information like the current value of objective function. The neighborhood $N(S)$ of a solution S is not a static set. Some neighbors in $N(S)$ are forbidden and the set of forbidden ('tabu') neighbors is stored in a short-term memory. At each local search iteration, the TS looks for a non tabu neighbor solution that gives maximum improvement to the objective function. Thus the introduction of tabu list eliminates the possibility of formation of a cycle. In the event of no improving moves, the TS method chooses the move that least deteriorates the objective function value. This feature helps the solution to escape from local optima.

The steps involved in a simple tabu search are described in the pseudo code [11] of Figure 1. An initial solution *S0* is computed and a tabu list T is initialized. The iterations are performed till the stopping criterion is achieved. The procedure *SelectBestNeighbor* returns the best non-tabu neighbor solution of the incumbent solution *S*. When the tabu list is full, then the oldest forbidden solution is removed from the list and the incumbent solution is entered into the list.

```
Procedure TabuSearch ()
01. Generate an initial solution S0 and set S ← S0
02. S* ← S , T ← ϕ; ;
03. while stopping criterion is not reached do
04.     S'← Select Best Neighbor ( N(S) \ T );
05.     if  f(S') < f(S*)  then
06.                 S* ← S';
07.     if |T|= Tabu list Size then
08.                 remove from T the oldest solution;
09.      T ← T ∪ S , S ← S';
10. end_while;
11. return S* ;
```

Figure 1 Pseudo code for tabu search method

B. Introduction to simulated annealing method

The theory of simulated annealing originated from the formation of crystals from liquids. The concept is based on the manner in which liquids freeze or metals recrystalize in the process of annealing. Let us consider how to coerce a solid into a low energy state. A low energy state usually means a highly ordered state, such as crystal lattice. To accomplish this, the material is annealed: heated to a temperature that permits many atomic rearrangements, then cooled slowly. As cooling proceeds, the system becomes more ordered and approaches a "frozen" stable state at a low temperature and the material freezes into a good crystal.

In physical chemistry, a low energy state usually means a highly ordered state, such as a crystal lattice; but it is also known that low temperature alone is not a sufficient condition for finding the ground state of the substance. When a substance is heated to a high energy state it permits many atomic rearrangements. If it is cooled in an un-controlled manner allowing the substance get out of equilibrium, it may form a glass or a crystal with defects.

The simulated annealing algorithm is a generalization of the Monte Carlo method. It transforms a poor, unordered solution into a highly optimized, desirable solution. This principle of simulated annealing technique with an analogous set of "controlled cooling" operations was used in the combinatorial optimization problems, such as minimizing functions of multiple variables, to obtain a highly optimized, desirable solution by Kirkpatrick [12]. The "balls and hills" diagram [13] in Fig. 2 illustrates an optimization problem in one dimension.

The cost surface is defined by including all the possible values of the cost function, taken over all legal configuration of *x*. In a normal gradient search method, the current configuration is perturbed only in the direction of reducing cost. This may results in the solution trapped in a local minimum. Simulated annealing allows perturbations to move uphill in a controlled fashion. Because each perturbation can transform one configuration into a worse configuration, it is possible to jump out of local minima and potentially fall into a more downhill path. However, because the uphill moves are carefully controlled; when we get closer to a good, final solution, we need not worry about getting out of it by an uphill move to some far worse one.

When the simulated annealing algorithm initially starts, the system is in a high energy state due to the random initial estimates of the coordinates of the sensor nodes. In each step of the algorithm, a sensor node is chosen sequentially from $m+1^{th}$ node to n^{th} node to be perturbed. The coordinate estimate $(\hat{x}_i, \hat{y}_i)$ of the chosen node is given a small displacement in a random direction. A new value of the cost function is calculated for the new location estimate. If the change in cost function, is less than or equal to zero, i.e., $\Delta(CF) \le 0$, then the perturbation is accepted and the new location estimate is used as the starting point of the next step.

```
Procedure SimulatedAnnealing ()
01. T= initial T ,  Δd = initial move distance
03. WHILE (stopping criteria is not reached)
05.    FOR i=1 to ( q×n )
06.        pick a node from the node set to perturb
08.        perturb the picked node by Δd in a random direction
08.        evaluate change in CF :  Δ(CF) = CF_new − CF_old
09.        if ( Δ(CF) ≤ 0 )
10.                accept this perturbation
12.        else
13.                pick a random probability RP uniformly
                   distributed in interval (0, 1)
14.                calculate the probability of acceptance
                   P(Δ(CF)) = exp(−Δ(CF) / T)
15.                if( RP ≤ P(Δ(CF)) )
16.                        accept this perturbation
17.                else
18.                        reject this perturbation and keep the
                           old configuration system
19.        end
20.        T = α×T, Δd = β×Δd
21.    end
```

Figure 2 Pseudo code for simulated annealing method

The case ($\Delta(CF) \ge 0$) is treated probabilistically:

the probability that the displacement is accepted is $P(\Delta(CF)) = \exp(-\Delta(CF)/T)$. Here T is known as the system "temperature" and P is a monotonically increasing function of T. A random number (*RP*) uniformly distributed in the interval (0, 1) is selected and compared with $P(\Delta(CF))$. If it is less than $P(\Delta(CF))$ then the perturbation is accepted and the new location estimate is used as the starting point of the next step. Otherwise, the perturbation is rejected and the original location estimate is kept.

Here temperature T is used as a control parameter to anneal the problem from a random solution to a good, frozen solution. Initially, the "temperature" T is set to a high value to permit aggressive, essentially random search of the configuration space. This could help the system jump out of local minimum. With the increase in the number of iterations, the decrease of system temperature results in decreasing probability of accepting a bad move. The cooling schedule $T = \alpha \times T, \alpha < 1$, is chosen to anneal the problem from a random solution to a good, frozen solution. The idea is to employ a cooling method to moderate the acceptance of uphill moves over the course of the solution. As the temperature cools, fewer uphill moves are allowed. The initial T and the cooling rate $\alpha < 1$ are determined empirically to give a good result. Initial "temperature" was set such that the probability of accepting a bad uphill move is about 80% in the beginning [14].

A move set is a set of allowable perturbation distances that will reach all feasible configurations and it should be easy to compute. Here the move set is chosen to be a random direction in the plane, multiplied by a small distance Δd in that direction. In order to control the generation of random moves at lower temperatures, we empirically restrict the change in distance as the temperature cools by introducing a shrinking factor $\beta < 1$, where $\Delta d = \beta \times \Delta d$.

Two criteria can be used to stop the SAL simulation: when the cost function *CF* is smaller than a predefined small number or when the predefined final temperature is reached.

IV. Localization Using Proposed Algorithm

A. First Step of the proposed algorithm

In the first step of the proposed algorithm, we use tabu search method to obtain an accurate estimate of the position of non anchor nodes. The position estimation of the coordinates of the target nodes involves the minimization of an objective function representing the error in locating the target nodes. The sum of squared range errors between the target nodes and neighboring anchor nodes is taken as the objective function for this problem [2].

$$f(x,y)=\sum_{i=1}^{N}\left(\sqrt{(x-x_i)^2+(y-y_i)^2}-c(t_i-t_0)\right)^2 \quad (2)$$

where

(x_i, y_i) : coordinates of i^{th} anchor node

(x, y) : coordinates of node to be estimated

t_i : time of arrival at i^{th} anchor node

t_0 : time of transmission of location query

N : number of neighboring anchor nodes

Our implementation of the TS method is as follows.

Step1: Selection of initial configuration

The centroid of the anchor nodes within the transmission range of the target node is a good initial estimate of the solution. The centroid is defined as Equation: (3)

$$(x_c, y_c)=\left(\frac{1}{N}\sum_{i=1}^{N}x_i, \frac{1}{N}\sum_{i=1}^{N}y_i\right) \quad (3)$$

Step2: Formation of tabu list

The tabu list is initialized with the centroid as its first entry.

Step3: Formation of neighborhood set

The initial estimate of the solution is given a disturbance which moves it to a new value. Though number of random perturbations is allowed, here they are restricted to eight movements that take the initial estimate to the neighbors $(x_c[\pm\Delta x], y_c[\pm\Delta y])$. Where $[\pm\Delta x]$ denotes the optional parameter that can be added to the *x* coordinates of the point to reach to a new neighbor.

Step4: Updating solution and tabu list

The solution is moved to a neighbor solution with minimum objective function and which is not in the tabu list. If the objective function value obtained for the best non-tabu neighbor solution is less than that for the best known solution, then the best known solution is updated with *S'* neighbor solution. The tabu list is appended with the solution that is already visited.

Step 5: Steps 3 and 4 are repeated by replacing (x_c, y_c) with the new value of the solution. This is continued till the objective function reaches below a predefined threshold value or the number of iterations crosses 100.

B. Refinement Step of the Proposed Algorithm

Optimizing the cost function (2) may not always result in an accurate localization, due to flip ambiguities. If a node's neighbors are placed in positions such that they are approximately on the same line, this node can be reflected across the line of best fit produced by its neighbors with essentially no change in the cost function. In Figure 3, the neighbors of node A are nodes B, C, D and E which are almost collinear and the node A could be flipped across the line of best fit of nodes B, C, D and E to location A' with almost no change in the cost function. In such situations, the particular node A is not uniquely localizable based on distance measurements. This phenomena is discussed in [15].

Let us define the sets N_i as a set containing all one hop neighbors of node *i* and $\overline{N}_i$ (the complement set of N_i) as a set containing all nodes which are not neighbors of node .

If R is the transmission range of the sensor node and the estimated coordinate of node $j \in N_i$ is such that $\hat{d}_{ij} < R$ in which $\hat{d}_{ij}$ denotes estimated destination between node *i* and *j*, then the node *j* has been placed in the wrong neighborhood of node *i*, resulting in both nodes *i* and *j* having each other as wrong neighbors.

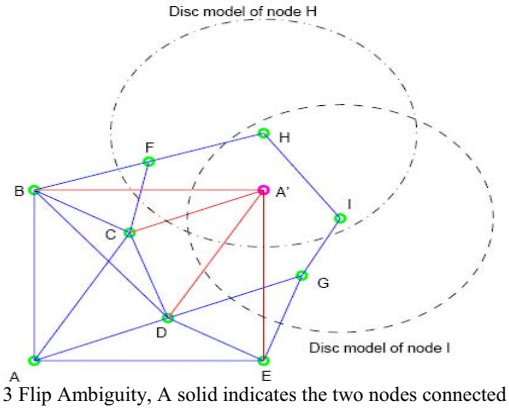

Figure 3 Flip Ambiguity, A solid indicates the two nodes connected by the line can communicate with each other directly.

That is, a node *i* could be placed into the wrong neighborhood either when the node *i* under consideration is flipped and moved in to a wrong neighborhood or another node *j*, which is a member of set N_i, has been flipped and estimated to be in set N_i. Here a disc model of radius *P* around each node is used to identify the nodes having the wrong neighborhood.

After the first phase of the localization, if a sensor node is in the correct neighborhood then it will be elevated to an anchor node. If the neighborhood of a sensor node is wrong, it will be placed in the set of nodes to be re-localized.

It is worth noting that, when the node density is low, there is possibility for a node to be flipped and still maintain the correct neighborhood. In situations like this, the node will be identified as uniquely localizable and thus elevated erroneously to an anchor node.

Refinement phase is an iterative algorithm. Given the initial location estimates of the first phase, the objective of the refinement phase is to identify the non-uniquely localizable nodes, which have either gone outside the boundary or having a wrong neighborhood and perturb them using the basic SAL principles with modified cost function to refine the results. If a node $j \in \overline{N}_i$ have been estimated such that $\hat{d}_{ij} < R$, then it has been placed in the wrong neighborhood and the minimum error due to the flip is $\hat{d}_{ij} - R$. The cost function for the refinement phase is modified to include this error term in order to introduce extra cost to a new estimate if it falls into the wrong neighborhood. This will influence the acceptance of the new estimation. This extra term helps push the new estimate away from the wrong neighborhood. Hence, the new localization problem can be formulated as the cost function below:

$$\min_{\substack{(x_i,y_i)\\ m<i\le n}} \sum_{i=m+1}^{n} \left(\sum_{j\in N_i} \left(\hat{d}_{ij} - d_{ij}\right)^2 + \sum_{\substack{j\in \overline{N}_i \\ \hat{d}_{ij}<R}} \left(\hat{d}_{ij} - R\right)^2 \right) \quad (4)$$

Since the proposed algorithm is centralized, it could have access to estimated locations and neighborhood information of all localizable nodes in the system.

In summary, the emphasis for first phase of the algorithm is to localize the uniquely localizable nodes accurately and give a good starting point to the refinement phase.

V. Simulation Results

In order to evaluate the performance of the proposed algorithm, many simulations were performed using visual studio .NET. A sensor network with a total of 200 nodes is simulated. Sensor nodes are uniformly distributed in a square region of 100×100. All the nodes are initialized with random coordinates within the boundary. The measured distance between the neighboring nodes, which is used in the cost function *CF*, is blurred by introducing a Gaussian noise into the true distance as shown in Equation (5)

$$d_{ij} = d_{ij}^{t} \times (1.0 + \eta \times NF) \quad (5)$$

Where d_{ij}^{t} and d_{ij} are true distance and measured distance respectively between the two nodes *i* and *j*. η is a Gaussian distributed random variable with 0 mean and variance 1. NF is the noise factor which controls the magnitude of the additionally introduced noise.

In our simulations noise factor is taken as 10%. The connectivity (average number of one-hop neighbors per node) is controlled by specifying the transmission range *R*.

The simulation results of applying just the first phase is shown in the Figures 4-5, and applying the second phase is shown in Figures 6-7.

VI. Conclusion

In this paper we have proposed and analyzed a new algorithm comprised of two phases for the localization of WSN using a short-term memory TS method and simulated annealing. From the performance analysis it is noted that in spite of its memory requirements, TS-based method has better convergence characteristics compared to simulated annealing based WSN localization proposed in [4]. Unlike the gradient search methods, TS method drives the solution search by escaping from local optima. The second phase of the algorithm is used to compensate the error caused by flip ambiguity by using simulated annealing method. Though we have considered only the 2D localization, the algorithm can be easily extended to 3D scenario.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, "A Survey on sensor Network," IEEE Communications Magazine, pp. 102-114, 2002.

[2] K. Yu and I. Oppermann, "Performance of UWB position estimation based on TOA measurements," in Join UWBST & IWUWBS, Kyoto, Japan, 2004, pp. 400-404.

[3] P. Biswas and Y. Ye, "Semidefinite programming for ad hoc wireless sensor network localization," Third International Symposium on Information Processing in Sensor Networks, pp. 46-54, 2004.

[4] A. Kannan, G. Mao, and B. Vucetic, "Simulated annealing based wireless sensor network localization," Journal of Computers, pp. 15-22, 2006.

[5] G. Mao, B. Fidan, and B. Anderson, "Wireless sensor network localization techniques," Computer Networks, vol. 15, pp. 2529-2553, 2007.

[6] C. Savarese and J. Rabaey, "Robust positioning algorithms for distributed ad-hoc wireless sensor networks," in The General Track: 2002 USENIX Annual Technical Conference, 2002, pp. 317-327.

[7] A. Savvides, H. Park, and M. B. Srivastava, "The bits and flops of the n-hop multilateration primitive for node localization problems," in International Workshopon on Sensor Networks Application, 2002, pp. 112-121.

[8] L. Doherty, K. pister, and L. E. Ghaoui, "Convex position estimation in wireless sensor networks," IEEE INFOCOM, vol. 3, pp. 1655-1663, 2001.

[9] T. C. Liang, T. C. Wang, and Y. Ye, "A gradient search method to round the semidefinite programming relaxation solution for ad hoc wireless sensor network localization," 5, 2004.

[10] F. Glover, "Tabu Search, Part I," ORSA Journal on Computing, vol. 1, pp. 190-206, 1989.

[11] S. L. Martins and C. C. Ribeiro, "Metaheuristics and applications to optimization problems in telecommunications," in Handbook of Optimization in Telecommunications, M. Resende and e. P. Pardalos, Eds. New-York: Springer Science + Business Media, 2006, pp. 103-128.

[12] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," Science, vol. 220, pp. 671-680, 1983.

[13] R. Rutenbar, "Simulated annealing algorithms: an overview," IEEE Circuits and Devices Magazine, vol. 5, pp. 19-26, 1989.

[14] K. Bryan, P. Cunningham, and N. Bolshakova, "Biclustering of expression data using simulated annealing," in CBMS. IEEE Computer Society, 2005, pp. 383–388.

[15] D. Moore, J. Leonard, D. Rus, and S. Teller, "Robust distributed network localization with noisy range measurements," in Second Int. Conf. on Embedded Networked Sensor Systems (SenSys 2004), 2004.

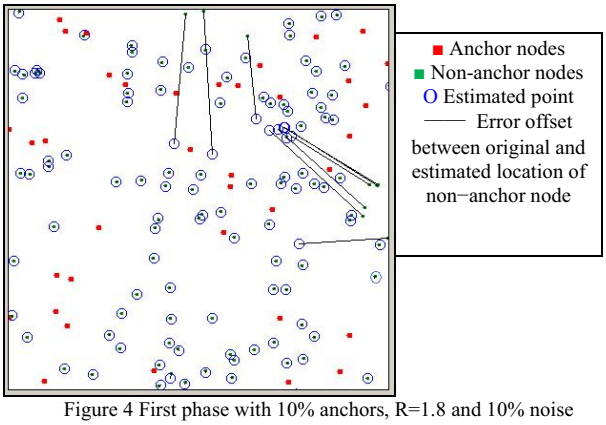

Figure 4 First phase with 10% anchors, R=1.8 and 10% noise

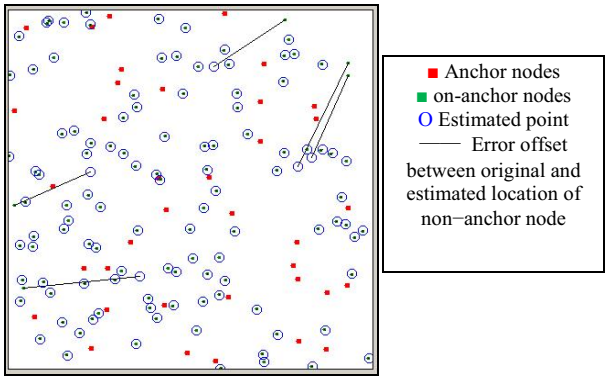

Figure 5 First phase with 10% anchors, R=2 and 10% noise

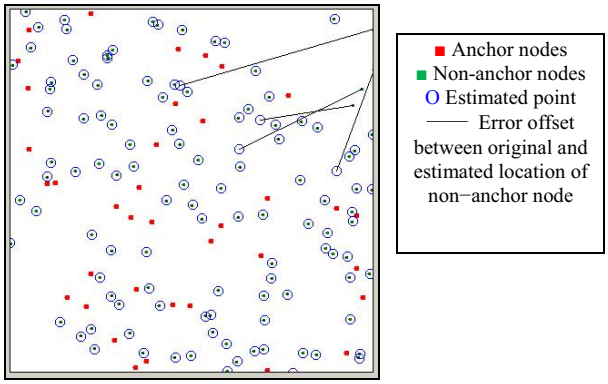

Figure 6 Second phase with 10% anchors, R=1.8 and 10% noise

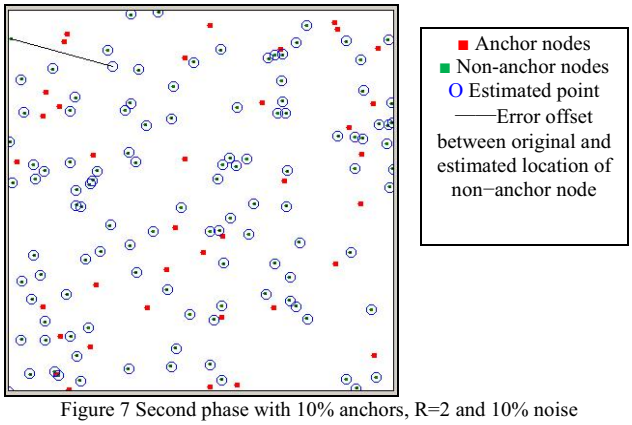

Figure 7 Second phase with 10% anchors, R=2 and 10% noise