Simulated Annealing based Wireless Sensor Network Localization with Flip Ambiguity Mitigation

Anushiya A Kannan*†, Guoqiang Mao*†and Branka Vucetic*
*School of Electrical and Information Engineering, University of Sydney, NSW 2006, Australia.
†National ICT Australia Limited[1],Locked Bag 9013, Alexandria, NSW 1435, Australia.
Email:{anushiya,guoqiang, branka}@ee.usyd.edu.au

***Abstract*— Accurate self-localization capability is highly desirable in wireless sensor networks. A major problem in wireless sensor network localization is the flip ambiguity, which introduces large errors in the location estimates. In this paper, we propose a two phase simulated annealing based localization (SAL) algorithm to address the issue. Simulated annealing (SA) is a technique for combinatorial optimization problems and it is robust against being trapped into local minima. In the first phase of our algorithm, simulated annealing is used to obtain an accurate estimate of location. Then a second phase of optimization is performed only on those nodes that are likely to have flip ambiguity problem. Based on the neighborhood information of nodes, those nodes likely to have affected by flip ambiguity are identified and moved to the correct position. The proposed scheme is tested using simulation on a sensor network of 200 nodes whose distance measurements are corrupted by Gaussian noise. Simulation results show that the proposed scheme gives accurate and consistent location estimates of the nodes and mitigate errors due to flip ambiguities.**

I. Introduction

In a sensor network, there will be a large number of sensor nodes densely deployed at positions which may not be predetermined. In most sensor network applications, the information gathered by these micro-sensors will be meaningless unless the location from where the information is obtained is known. This makes localization capabilities highly desirable in sensor networks. A main problem in wireless sensor network localization based on distance measurements is the flip ambiguity, which introduces large errors in the location estimates. Flip ambiguity arises when a node's neighbors are placed nearly collinear such that the node can be reflected across the line connecting it's neighbors while satisfying the distance constraint, or in the case of erroneous distance measurements.

This paper proposes a two-phase simulated annealing based localization algorithm (SAL), where an initial location estimate is obtained in the first phase using the simulated annealing technique and the large error due to flip ambiguity is mitigated in the refinement phase using neighborhood information of nodes. The proposed algorithm is implemented in a centralized architecture, where all nodes send their measurements to a central station for localization. Generally speaking, a distributed architecture will improve scalability and reduce complexity of the algorithm. However in some applications such as monitoring patients and assisting disabled patients in the health sector; monitoring and controlling homes and cities; monitoring bush fire, water quality etc. in the environment; monitoring humidity, temperature etc. in the precision agriculture, a central system exists, which gathers information from all nodes hop-by-hop and make decisions accordingly.In such applications with a centralized architecture, it may be more convenient to implement a centralized localization algorithm. The feasibility of nodes communicating their information to a central station has been demonstrated in [1].

The rest of the paper is organized as follows: section II summarizes related work in the area; section III presents a brief overview of the simulated annealing technique; section IV describes our proposed SAL approach where subsection IV-A introduces the basic SAL technique and describes the first phase of the proposed algorithm and subsection IV-B talks about the flip ambiguity problem and describes the refinement phase of the proposed algorithm; and section V presents simulation results. We conclude this paper in section VI, together with intended future work.

II. Related Work

Many researchers have approached the localization problem from different perspectives. Here we focus on localization methods based on distance measurements. Generally it is assumed that there are a number of anchor nodes in the sensor network. The locations of anchor nodes are known and they are used to fix the local coordinate system. However due to constraints on cost and complexity, most sensor nodes have unknown locations, which are to be estimated by the localization algorithm.

Niculescu [2] proposed a distributed, hop-by-hop localization method (APS) based only on the connectivity information. Savarese [3] improved Niculescu's algorithm by introducing a refinement phase. In the refinement phase, distance measurements between neighbors are used to improve localization accuracy. To prevent error accumulation, Savvides [4] used least squares estimation with Kalman filter to simultaneously locate the positions of sensor nodes.

[1]National ICT Australia is funded by the Australian Government's *Backing Australia's Ability* initiative, in part through the Australian Research Council.

Doherty [5] approached the problem using convex optimization based on semi-definite programming. The connectivity of the network is represented as a set of convex localization constraints for the optimization problem. Biswas [7] extended this technique by taking the non-convex inequality constraints and relaxed the problem to a semi-definite program. Tzu-Chen Liang improved Biswas's method further by a gradient search technique [6].

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

III. Simulated Annealing Technique

Simulated Annealing (SA) is a technique for combinatorial optimization problems, such as minimizing functions of multiple variables. It is a generalization of the Monte Carlo method. The concept is based on the manner in which liquids freeze or metals recrystalize in the process of annealing.

The simulated annealing algorithm exploits an analogy between the way in which a metal cools and freezes into a minimum energy crystalline structure (the annealing process) and the search for a minimum in a more general system. 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 many variables, to obtain a highly optimized, desirable solution by Kirkpatrick [8].

In a normal gradient search method, the current configuration is perturbed only in the direction of reducing cost. Each new perturbation moves to a configuration downhill from the previous one. This may results in the solution being trapped in a local minima. 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.

IV. Simulated Annealing based Localization

The location estimation problem has a natural analogy with the simulated annealing algorithm. Consider a sensor network of m anchor nodes with known locations and $n-m$ sensor nodes with unknown locations. For simplicity, let the nodes lie on a plane such that node i has location (x_i,y_i). Initially all the sensor nodes are initialized with random locations (x_i,y_i) within the boundaries of the sensor network. Let us define a graph $G = (V(G), E(G))$, where $V(G)$ is the set of all nodes in the sensor network and $E(G)$ is the set of all links between one-hop neighbors. If G is a disconnected graph such that a component of G, $G1 = (V(G1), E(G1))$ has the vertex set

T = initial T
Δd = initial move distance
WHILE (final T is not met AND cost function CF is not acceptably small)
{
FOR $i = 1$ to $(q * N)$
{
pick a node from the node set to perturb
DO p times
{
perturb the picked node by Δd in a random direction
evaluate the change in cost function $\Delta(CF) = CF_{new} - CF_{old}$
if $(\Delta(CF) \leq 0)$
//downhill move => accept it
accept this perturbation and update the configuration system
else
//uphill move => accept with a probability
pick a random probability RP uniformly distributed in interval $(0, 1)$
calculate the probability of acceptance $P(\Delta(CF)) = exp(-\Delta(CF)/T)$
if$(RP \leq P(\Delta(CF)))$
accept this perturbation and update the configuration system
else
reject this perturbation and keep the old configuration system
}
}
$T = \alpha * T$
$\Delta d = \beta * \Delta d$}

Fig. 1. SAL Algorithm

$V(G1)$ which does not have three or more anchors in $G1$, then all the sensor nodes in the subgraph $G1$, i.e all members of vertex set $V(G1)$, is non-localizable and they are removed from the localization system.

A. First Phase of the Proposed Algorithm

In the first phase, simulated annealing is used to obtain an accurate estimate of location of the localizable sensor nodes using the distance constraints. Let us define the set N_i as a set containing all one hop neighbors of node i. The localization problem can be formulated as:

$$\min_{\substack{(x_i,y_i)\\ m<i\leq n}} \underbrace{\sum_{i=m+1}^{n} \sum_{j\epsilon N_i} (\hat{d_{ij}} - d_{ij})^2}_{CF} \tag{1}$$

where d_{ij} is the measured distance between node i and it neighbor node j; $\hat{d_{ij}} = \sqrt{(\hat{x_i} - \hat{x_j})^2 + (\hat{y_i} - \hat{y_j})^2}$ is the estimated distance; ($\hat{x_i}$,$\hat{y_i}$) and ($\hat{x_j}$, $\hat{y_j}$) are the estimated coordinates of node i and its one hop neighbor node j respectively.

The cost function CF shown in (1), represents the quantitative measure of the "goodness" of the coordinate estimate. Our aim is to optimize the cost function using simulated annealing technique to get the optimal location estimate without being trapped in a local minimum. The structure of the simulated annealing algorithm is shown in Fig.1.

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 i 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 $\Delta(CF)$, is less than or equal to zero, i.e. $\Delta(CF) \leq 0$, then the perturbation is accepted and the new location estimate is used as the starting point of the next step.

$$\Delta(CF) = CF_{new} - CF_{old} \tag{2}$$

The case $(\Delta(CF) > 0)$ is treated probabilistically: the probability that the displacement is accepted is $P(\Delta(CF)) = exp(-\Delta(CF)/T)$. Here T is a control parameter, which by analogy with the SA 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. At a high "temperature" the probability of accepting a large uphill move is high. 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 $T = \alpha * T, \alpha < 1$, 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. Most of the uphill moves are allowed at higher temperatures. 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 [10].

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 $\Delta d = \beta * \Delta d$.

To get the optimum performance out of the simulated annealing technique, it is necessary to cool carefully and slowly, allowing it to come to thermal equilibrium at each temperature. At each temperature, $p*(q*N)$ perturbations are performed in order to get the system into equilibrium in that particular temperature. Here p is the number of perturbations given to a particular sensor node at a temperature and $(q*N)$ is the number of sensor nodes perturbed at a temperature where N is the number of nodes in the sensor network and q is a reasonably large number to make the system go into thermal equilibrium.

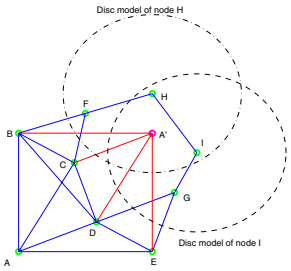

Fig. 2. Flip Ambiguity

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.

B. Refinement Phase of the Proposed Algorithm

Optimizing the cost function (1) 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 it's neighbors with no change in the cost function, then it is said to have flip ambiguity. In Fig.2, 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 no change in the cost function. In such situations, the particular node is not uniquely localizable based on distance measurements only. This phenomena of flip ambiguity is discussed in [11]–[13]. But a good method to solve this problem is yet to be found. In order to address this problem, a second round of optimization is performed only on those nodes that are likely to have flip ambiguity. We should notice in Fig. 2 that the flipped position A' has gone into the wrong neighborhood of nodes H and I. In a sensor network with a medium to high node density, there is a high chance that a node's location estimate, which has been flipped, will fall into the wrong neighborhood of other nodes or outside the sensor network boundary. Based on this observation, a second phase of optimization is performed to identify location estimates of nodes which are likely to have flipped and move them to the correct locations.

Let us define a complement set $\overline{N_i}$ of the set N_i as a set containing all nodes which are not neighbors of node i. If R is the transmission range of the sensor node and a node $j \in \overline{N_i}$ is estimated such that $\hat{d_{ij}} < R$, 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. That is, a node i could have 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 $\overline{N}_i$ has been flipped and estimated to be in set N_i. Here a disc model of radius R around each node is used to identify the nodes having the wrong neighborhood.

After the first phase of the localization, if the neighborhood of a sensor node is correct then it will be elevated to an anchor node. Refinement phase will not perturb these elevated anchors. If the neighborhood of a sensor node is wrong, it will be identified as non-uniquely localizable node and placed in the set of nodes to be re-localized using the refinement phase. For example in Fig. 2, all nodes will be localized accurately in the first phase except A. Because of the flip of node A, nodes A, H and I all will be seen as in the wrong neighborhood and placed in the set of nodes to be re-localized using the refinement phase. The node A will be localized more accurately in the refinement phase at the cost of the nodes H and I being localized slightly less accurately than the first phase.

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 which is used to improve upon and refine the location estimates generated by first phase. 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. With these information, the localization problem in the refinement phase can be formulated as:

$$\min_{\substack{(x_i,y_i)\\ m<i\leq n}} \underbrace{\sum_{i=m+1}^{n} \Big(\sum_{j\epsilon N_i} (\hat{d_{ij}} - d_{ij})^2 + \sum_{\substack{j\in\overline{N}_i\\ \hat{d_{ij}}<R}} (\hat{d_{ij}} - R)^2\Big)}_{CF} \quad (3)$$

The cost function CF in the refinement phase is different to that used in the first phase of the algorithm. Since the proposed algorithm is implemented in a centralized architecture, 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 SAL is to localize the uniquely localizable nodes accurately and give a good starting point to the refinement phase of SAL. Refinement phase of SAL focuses on increasing the accuracy of the localization by optimizing only the non-uniquely localizable nodes with the new cost function.

TABLE I

TRANSMISSION RANGE VS. THE CONNECTIVITY

Transmission Range	1.1	1.2	1.4	1.6	1.8	2.0
Ave. Connectivity	6.86	8.19	10.87	13.96	17.81	21.36

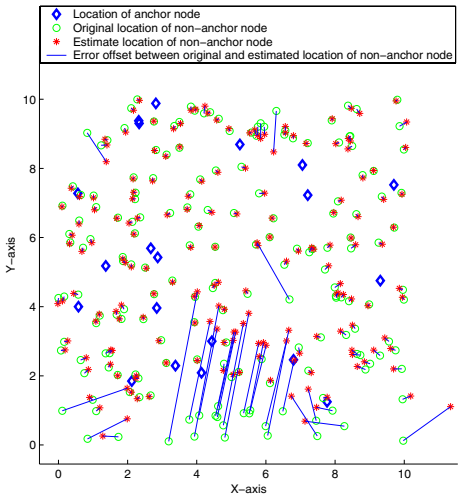

Fig. 3. First phase with 10% Anchor & Transmission range of 1.3

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 10×10. All the sensor nodes are initialized with random coordinates within the boundary. The values of p and q in Fig.1 are chosen as 10 and 2 respectively. 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 Eq.4.

$$d_{ij} = d_{ij}^t * (1.0 + \text{Gaussian Noise}() * \text{Noise Factor}) \quad (4)$$

where d_{ij}^t and d_{ij} are true distance and measured distance respectively between the two nodes i and j.

In our simulations noise factor is taken as 10%. The Gaussian noise has a mean of 0 and a standard deviation of 1. The connectivity (average number of one-hop neighbors per node) is controlled by specifying the transmission range R. A relationship of transmission range vs. connectivity is tabulated in Table I. To allow for easy comparison between different scenarios, errors in location estimates are normalized by the transmission range. In Figs.3 and 4 the true and estimated sensor locations are shown and an error offset line has been drawn between the true and estimated locations. Fig.3 shows a location estimate after the first phase of the proposed algorithm and Fig.4 shows the location estimate after the refinement phase. It is shown in the figures that the refinement phase mitigates the problem of flip ambiguity and improves the location estimates.

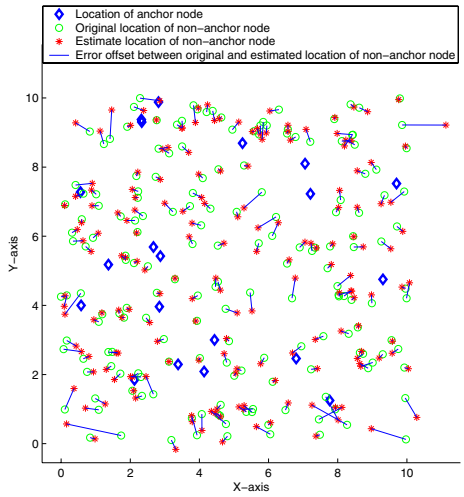

Fig. 4. Refinement phase with 10% Anchor & Transmission range of 1.3

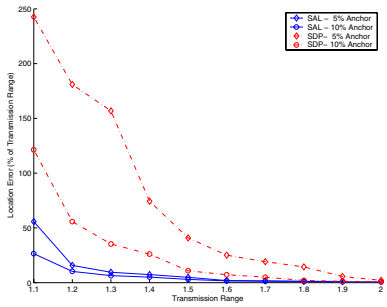

Fig. 5. Location error of uniformly distributed sensor nodes

It was reported that semi-definite programming with gradient search localization (SDPL) proposed in [6] gives much better performance than the other methods in the literature. Thus we compare the performance of our proposed SAL algorithm with SDPL. Fig.5 shows the simulation results with varying transmission range while having 5% anchor nodes and having 10% anchor nodes respectively. For each point in Fig.5, ten simulations are performed with different random seeds and the average error is shown. The location error is calculated as in (5). It is reported in percentage, normalized by the transmission range.

$$\text{location error} = \frac{1}{(n-m)} * \frac{\sum_{i=m+1}^{n} (x_i - \hat{x_i})^2 + (y_i - \hat{y_i})^2}{R^2} * 100\% \quad (5)$$

where $(\hat{x_i},\hat{y_i})$ is the estimated location of sensor node and (x_i,y_i) is the true location of sensor node.

In SAL when the connectivity is 14 or above, the mean square location error goes below 1%, no matter how many anchor nodes are present. As shown in Fig.5, the proposed algorithm performs much better than SDPL.

VI. Conclusion and Future Work

In this paper we presented a simulated annealing based localization algorithm which mitigate the flip ambiguity problem. The proposed algorithm is divided into two phases. The first phase uses the simulated annealing technique to obtain an accurate estimate of the nodes' location. The second phase is focused on the nodes that are likely to have flip ambiguity problem. It uses the neighborhood information to identify these nodes and move the nodes to the correct location. Simulations were performed which demonstrated that the proposed algorithm gives better accuracy than the semi-definite programming localization. It was also shown that the proposed algorithm does not propagate error in localization.

The proposed flip ambiguity mitigation method is based on neighborhood information of nodes and it works well in a sensor network with medium to high node density. However when the node density is low, it is possible that a node is flipped and still maintains the correct neighborhood. In this situation, the proposed algorithm fails to identify the flipped node. It is part of our future work to develop a more robust technique to identify nodes that are likely to have flip ambiguity problem.

Still another direction for our future work is investigating implementation of the proposed algorithm in a distributed architecture. This shall improve scalability of the algorithm.

References

[1] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for networked sensors," *Operatin Systems Review 2000*, vol. 34, pp. 93–104.

[2] D. Niculescu and B. Nath, "Ad hoc positioning system (aps)," in *IEEE GLOBECOM 2001*, vol. 5, pp. 2926–2931.

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

[4] 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 Sensor Networks Application 2002*, pp. 112–121.

[5] L. Doherty, K. pister, and L. El Ghaoui, "Convex position estimation in wireless sensor networks," in *IEEE INFOCOM 2001*, vol. 3, pp. 1655–1663.

[6] 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 localisation," Sanford University, formal report 5, 2004. [Online]. Available: `http://www.stanford.edu/~yyye/formal-report5.pdf`

[7] P. Biswas and y. Ye, "Semidefinite programming for ad hoc wireless sensor network localization," in *Information Processing in Sensor Networks, 2004. IPSN 2004*, pp. 46 – 54.

[8] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[9] R. Rutenbar, "Simulated annealing algorithms: an overview," *IEEE Circuits and Devices Magazine*, vol. 5, no. 1, pp. 19–26, 1989.

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

[11] D. Moore, J. Leonard, D. Rus, and S. Teller, "Robust distributed network localization with noisy range measurements," in *Proc. 2nd ACM SenSys*, 2004.

[12] T. Eren, D. Goldenburg, W. Whiteley, Y. Yang, A. Morse, A. B.D.O, and B. P.N, "Rigidity, computation, and randomization in network localisation," in *IEEE INFOCOM 2004*, vol. 4, pp. 2673 – 2684.

[13] D. Goldenburg, W. Krishnamurthy, A.and Maness, Y. Yang, and A. Young, "Network localization in partially localizable networks," in *IEEE INFOCOM 2005*.