This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE SECON 2006 proceedings.

Consistency Error Modeling-based Localization in Sensor Networks

Jessica Feng
Computer Science dept.
University of California, Los Angeles
Los Angeles, CA USA
jessicaf@cs.ucla.edu

Miodrag Potkonjak
Computer Science dept.
University of California, Los Angeles
Los Angeles, CA USA
miodrag@cs.ucla.edu

***Abstract*—We have developed a new error modeling and optimization-based localization approach for sensor networks in presence of distance measurement noise. The approach is solely based on the concept of consistency. The error models are constructed using non-parametric statistical techniques; they do not only indicate the most likely error, but also provide the likelihood distribution of particular errors occurring. The models are evaluated using the learn-and-test techniques and serve as the objective functions for the task of localization. The localization problem is formulated as task of maximizing consistency between measurements and calculated distances. We evaluated the approach in (i) both GPS-based and GPS-less scenarios; (ii) 1-D, 2-D and 3-D spaces, on sets of acoustic ranging-based distance measurements recorded by deployed sensor networks. The experimental evaluation indicates that localization of only a few centimeters is consistently achieved when the average and median distance measurement errors are more than a meter, even when the nodes have only a few distance measurements. The relative performance in terms of location accuracy compare favorably with respect to several state-of-the-art localization approaches. Finally, several insightful observations about the required conditions for accurate localization are deduced by analyzing the experimental results.**

Keywords-consistency; error modeling; location discovery

I. INTRODUCTION

The *localization (location discovery or LD)* problem can be defined in the following way. A total of N nodes, K of which ($K<<N$) have the exact information about their positions. The measured distances, which are subject to errors, between M pairs of nodes are also available. The goal is to conclude the location (x_i, y_i) of each unknown location node i in such a way that $L(x_{ri} - x_i, y_{ri} - y_i)$ is minimized, where (x_{ri}, y_{ri}) is the real location of i. Usually the targeted L is L_1, L_2, or L_∞.

It has been proven that the localization problem is NP-complete [1]. It is also easy to see that the localization problem belongs to the class of nonlinear programs. A great variety of centralized algorithms (executed at a single place with the availability of the complete information about all measurements) and localized algorithms (executed by multiple nodes simultaneously and/or consecutively where each node has limited information provided by its neighbors) have been proposed. They range from iterative linearization and convex programming to conjugate direction-based and multiresolution search. [2] provide comprehensive surveys of state-of-the-art positioning designs and signal processing techniques. However, the effectiveness of these algorithms is constrained by the accuracy of the error model. There is a wide spectrum of available error models ranging from closed form parametric models to sophisticated kernel estimation-based non-parametric models. Nevertheless, none of them is a-priori applicable in new environments. The following small example shown in Figure 1 demonstrates the importance of the correct error model.

Consider 10 nodes $N_1,...,N_{10}$. We assume that the locations of the first nine nodes are available and error free. The topology of these 10 nodes is taken from a deployed network. The distances between the nodes are estimated based on the time-of-arrival of the acoustic signals. The traveling time of the acoustic signals is multiplied with the speed of the sound to estimate the distances between nodes – the *measured distances* [3][4]. Table 1 contains the information about the locations of the nine nodes (the second column); the *real/correct distances* obtained using the distance formula given the real locations of the nodes (the third column); the measured positions on two different days (the fifth and the sixth columns – STAT1 and STAT2). All measurements are in meters. In addition, the forth column shows the *simulated distances* generated under the widely used assumption of Gaussian noise model [5][6] on top of the real distances.

The goal is to locate N_{10} using the measured/simulated distances. We obtain the solution using the exhaustive search and following the maximum likelihood principle. Table 2 shows the results in term of location error, i.e. $(x_{r10} - x_{10}, y_{r10} - y_{10})$. The three rows indicate which set of measured/simulated distance measurements is used to derive N_{10}'s location (i.e. which type of error is in the distance measurements), and the four columns indicate the type of errors targeted by the maximum likelihood (i.e. the error model used as the optimization target). We see that when the correct type of errors is targeted, low location discrepancy is achieved, indicated by the bold italic numbers in Table 2. The average location error is between 1 and 3.3cm although some individual measurements have errors of more than 40m. However, when the errors in measurements and the optimization targeted error model do not match, the location error increase significantly. For example, when the Gaussian error model is assumed for the minimization of errors on the actually collected data – STAT1, the location error is more than 8m (8.179m). Even when the model obtained on one day is used as the optimization objective on another day, the

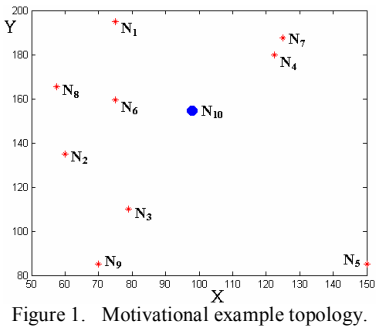

Figure 1. Motivational example topology.

Table 1. The distance measurements information.

ID	LOCATION	REAL	GAUSSIAN	STAT 1	STAT 2
N_1	(75, 195)	45.893	45.791	56.697	44.193
N_2	(60, 135)	42.5	43.432	42.895	42.043
N_3	(79, 110)	48.654	78.066	49.008	39.964
N_4	(122.5, 180)	35.355	35.294	34.355	42.139
N_5	(150, 85)	87.5	86.362	56.988	87.479
N_6	(75, 159.4)	22.926	53.285	23.001	27.077
N_7	(125, 187.5)	42.573	42.938	43.837	41.992
N_8	(57.5, 165.4)	41.337	42.831	41.111	49.604
N_9	(70, 85)	75.208	71.427	87.449	74.574

Table 2. Solutions resulted using different error models (columns) based on different sets of measurements (rows).

	GAUSSIAN	STAT 1	STAT 2	CONSISTENCY
GAUSSIAN	***0.0208***	7.993	4.258	0.0302
STAT 1	8.179	***0.0117***	5.275	0.0215
STAT 2	7.658	6.042	***0.0303***	0.0310

resultant location error still stays above 5m (6.042m and 5.275m). Therefore, we conclude that unless an accurate error model with respect to the measurements is targeted, accurate location discovery is not possible.

However, a simple condition of pair-wise consistency easily resolves this problem, at least for the example shown in Figure 1. We say that a pair of measurements is *pair-wise consistent* if the longer measurement corresponds to the longer real distance. The formal definition of consistency is stated in Section 4. The last column in Table 2 shows the location errors yield using the error model derived based on the concept of consistency. Regardless of what type of errors is in the distance measurements, the location error of N_{10} is always around 3cm. The final observation is that maximizing percentage of consistent measurements can be easily mapped to nonlinear function minimization problem and solved using standard software.

II. RELATED WORK

In this section, we survey various location discovery algorithms. Location discovery refers to the task where all the unknown-location nodes seek to determine the relative and/or absolute position using the measured distance between different nodes. Such a distance can be measured by approaches include acoustic ranging methods [3][8][10], RSSI and RF proximity estimation [4][5], as well as algorithmic techniques [3][6]. In this paper, the set of distance measures we used as the demonstrative example was collected based on the line-of-sight acoustic signals [8][10].

Location discovery algorithms can be either centralized or localized [11]. Centralized algorithms assume that all the measured distances are forwarded to the center node, which then computes the location of each node using such information. Localized algorithms do not require the existence of the center node and allows each node to compute its position based on its local information by atomic multilateration, a method to estimate the location of a node if it is within the communication range of at least three beacons [12]. Iterative multilateration algorithm uses atomic multilateration as the primitive and treats an unknown node as a beacon once its location is resolved [13].

There are in general two different scenarios the location discovery problem is solved under. One of which assumes the measured distances between communicating nodes are available. Some of the recent work which are based on this assumption include [14][15][16]. Sheng and Hu [15] present a localization approach based on the acoustic energy decay model, where the acoustic energy decays inverse of distance square under the some mild conditions. This energy based localization problem is then solved by combining Maximum Likelihood (ML) estimation with Expected Maximization (EM) solution and projection solution. In addition, they also derived the Cramer-Rao Bound (CRB) for sensor deployment analysis. Niculescu and Nath [16] propose a localization approach which is based on the basic idea of distance vector routing using only a fraction of beacons, with the assumption that each sensor node has some combination of ability to measure range, angle of arrival (AOF), orientation. They propose a lower bound for positioning error for a range/angle free algorithm, and examine the error characteristics of various classes of multihop ad-hoc positioning systems (APS) algorithms. The localization method proposed by Galstyan et al. [17] is distributed and on-line, which means the localization process is conducted simultaneously with an application task. Sensor nodes use their geometric constraints induced by radio connectivity and sensing to decrease the uncertainty of positions. The performance of the algorithm is compared with the centralized (convex) programming. In addition to static networks, Hu and Evens [15] introduce the sequential Monte Carlo localization method for mobile networks, which exploit mobility to improve the accuracy and precision of positioning. The approach does not require any additional hardware and has competitive results when compared to static localization methods. A comprehensive study of the fundamental limitations and location accuracy bound for mobile positioning is presented in [18].

The second scenario which location discovery is solved under does not put any requirement on the availability of measured distances [19][20]. He et al. [19] propose a range-free localization approach which performs the best when an irregular radio pattern and random node placement are considered. The algorithm is called area-based, which has two phases: i) isolating the environment into triangular regions between beacon nodes; ii) by considering whether a node's presence inside or outside of these triangular regions allows a node to narrow down the area in which it can potentially reside. The algorithm is demonstrated in conjunction with the routing and the tracking problems. Shang et al. in [20] presents a

localization method that uses the connectivity information (i.e. who is within the communication ranges of whom) to derive the positions of the unknown sensors.

III. PRELIMINARIES

A. The Distance Measurements

We construct the statistical error models and conduct location discovery on sets of distance measurements that are collected using the acoustic signal detection-based ranging techniques. The number of deployed sensor nodes varies from 79 to 93, with the average being 90. The sensor nodes are custom designed based on an SH4 microprocessor running at 200MHz (Figure 2(a)). The nodes are deployed at the Fort Leonard Wood Self Healing Minefield Test Facility, which has size 200m x 50m. The radio signal (communication) range is about 50m. Figure 2(b) shows an example of deployment topology. Each node is equipped with four independent speakers and microphones as the ranging tool. The distance between two nodes is obtained by timing the arrival of the acoustic signals [8]. Each node in the network takes turns to transmit the acoustic signals, all the nodes that receive the signals record the time of arrival and convert the time of flight to distance in meters. There are total 33 sets of distance measurements collected over the course of few days; each set consists of one round of acoustic signal transmission by all the nodes. For the sake of simplicity, we demonstrate the algorithms and techniques on a randomly selected set of measurements, and we present the results for ten other randomly selected data sets in Section 5. The details on the experimental setup and the acoustic detection scheme used can be found in [8][10].

From the communication point of view, we distinguish two types of the communications between a pair of nodes: i) exchange of the acoustic signals for the purpose of distance ranging; ii) transmission and reception of radio signals (in terms of bytes) for the purpose of exchanging information. More specifically, we denote L_i as a set of nodes that receive node i's acoustic signals, therefore can estimate the distances between themselves to node i. Similarly, C_i denotes a set of nodes that receive the radio signals from i. We assume that the acoustic signal range (ASR) is independent from the radio signal range (RSR), which means that it is possible for a node i to have the distance estimate to another node j (i has received j's acoustic signals), while i can not exchange information with j (j is out of i's radio signal range), and vise versa. Furthermore, it is not necessary that all nodes in the network have the same ASR and RSR properties. This is a more realistic reflection of the actual deployed networks.

B. Location Discovery

The location discovery problem can be formally stated as follows. In a k dimensional space, when we consider the homogeneous case where two sensor nodes i (x_{1i}, x_{2i},…,x_{ki}) and j (x_{1j}, x_{2j},…,x_{kj}) have measured distance d_{ij}, exactly one equation of the form of Equation (1) can be written where ε_{ij} denotes the discrepancy between the calculated distance and the measured distance.

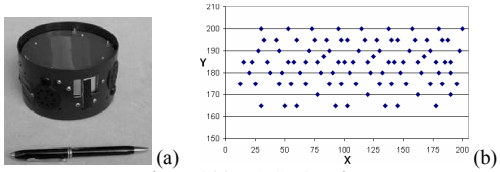

Figure 2(a). A SH4 node.
Figure 2(b). An example of the deployment topology.

$$\varepsilon_{ij} = \sqrt{\sum_{l=1}^{k}(x_{li} - x_{lj})^2} - d_{ij} \quad (1)$$

After a set of equations that correspond to the pairs of nodes that have measured distances are written, where the unknown variables being the coordinates of the unknown nodes, the system of equations is then linearized and fed to a linear optimization mechanism. [13] provides a detailed procedure of how the system of equation is linearized.

We formulated the location discovery problem in terms of a nonlinear function minimization instance where the objective function F has the form expressed in Equation (2). Function M can take the form of L_1, L_2, L_∞ norms (F is subject to minimization), or the Gaussian distribution with various variances or the statistical error model constructed using the kernel density estimation technique (F is subject to maximization). In our study, M is the pair-wise consistency-based error model.

Nonlinear programming is a direct extension of linear programming where the linear objective function is replaced by the nonlinear ones. Nonlinear programming has advantages in terms of computing power and formulation flexibility. The most important reason why we formulated the localization problem as a nonlinear programming is due to the NP-completeness of the localization problem [1].

$$F = M(\varepsilon_{ij}) \quad (2)$$

where $\varepsilon_{ij} = \sqrt{\sum_{l=1}^{k}(x_{li} - x_{lj})^2} - d_{ij}$

for pairs of nodes i&j that have measured distance d_{ij}.

IV. OFF-LINE LOCALIZATION

Statistical models that predict the variable of high importance from an easy to measure variable are off great importance in sensor networks and many other domains. The models can be used to make a number of decisions during the realization of many applications. We define consistency as the pair-wise relationship between two pairs of predicting and predicted variables. More specifically, two pairs $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$ are consistent with respect to each other if and only if (Equation (3)).

$$((x_1 \geq x_2) \Rightarrow (y_1 \geq y_2) \vee (x_1 \leq x_2) \Rightarrow (y_1 \leq y_2)) \quad (3)$$

It is easy to see that consistency is the necessary and sufficient requirement to make correct decisions when selecting between two or more options for the predicted variable from the predicting variables. In addition, stability of a particular data set for constructing error models is often well captured by high consistency.

In off-line error modeling, the predicting variable is the measured distances; the predicted variable is the corresponding real distances. In our study, the real distances are calculated using the distance formula given the correct coordinates of the nodes for all measured distances. The models obtained in this section have multiple usage: i) serves as a reference for model comparison; ii) validation of solutions of other methods (e.g. on-line LD); iii) if only measurements from beacons are available, the model construction techniques still can be applied to the limited set of measurements in order to evaluate possible solutions; iv) serves as a starting framework for other data sets, if similar environments are observed. A more complex and realistic scenario where no real distances are available is discussed in detail in [21]. Conducting regressions (e.g. monotonic regression) is the first step of constructing consistent error models. In this section, our first objective is to develop model that relies solely on the notion of consistency. This requirement has two ramifications: i) any arbitrary two points belong to the regression curve satisfy the consistency requirement with respect to each other; ii) all points belongs to the regression curve that maps the predicting to the predicted variable are maximally consistent with respect to all the available measurements.

Therefore, the regression function is either monotonically non-decreasing or monotonically non-increasing. By mapping the problem of developing monotonic and consistent error models into discrete domain and further into the graph theoretic framework, we are able to develop provably optimal algorithms of polynomial complicity. This technique is versatile in the sense that it can be adopted to satisfy many additional requirements such as the restriction of the maximum and the minimum slopes of the curve. In addition, the algorithm can be applied not only to regression, but also to the kernel density estimation, where not only the most likely value of the predicted variable given a predicting value is derived, but also the likelihood of the predicted variable having a particular arbitrary value.

A. Regression

In this section, we introduce the technique for off-line consistency-based regression, which is the first step towards constructing the consistency-based error model. We start by stating the procedure of mapping the continuous instance to the discrete domain in order to enable the application of the imitable graphical theoretic. After that, the instance is transformed to a graph format where finding the most consistent monotonic regression function is equivalent to finding the shortest path of the graph. The shortest path problem is solved using a simplified dynamic programming-based Dijkstra's shortest path algorithm. We conclude the section by analyzing the complicity of the algorithm and presenting the regression accuracy using the standard learn-and-test technique [7]. Furthermore, we introduce the modified versions of the algorithm to achieve different objectives such the unimodularity of the regression function, enhancement of the robustness, and minimum control complexity of the regression function.

The input of the regression is a set of pairs of measured distances and their corresponding real distances. Figure 3 shows an instance of 750 such pairs, where the x and y axes of each data point indicate the measured and the corresponding real distance respectively. We first establish how well the data set is suitable for modeling in a quantitative way by examining the consistency among data points. For this particular set of data points, the consistency is 0.844; and 10 data points need to be excluded for consideration in order to achieve a consistency 0.967; 20 points need to be excluded in order to achieve a consistency 0.981. Clearly, only a few of inconsistent data points are contributing to the inconsistency. We conclude that the data set is suitable for regression and modeling. In order to transform the instance from continuous to discrete domain, we superimpose an uniform distance grid on top of space where the data points are placed. It is important to note that there are many different procedures to superimpose a grid, such as based on an uniform number of data points or an uniform relative error in each grid. For the sake of simplicity, in our description, we will reside our attention on the uniform distance grid. After the transformation, all the data points within each grid cell are treated and weighted equally. The granularity of the grid can be either specified by the user or statistically determined by the procedure introduced in the later section. After the grid is specified, the number of data points in each grid cell is counted (Figure 3). The goal is to determine a regression function that goes though grid cells in such a way that the total inconsistent data points in these grid cells is the minimum. Therefore, the regression function is most consistent with respect to all the data points. After counting the number of points in each cell, the next step is to calculate the inconsistency cost/charge of each grid cell with respect to all other grid cells according to the previous definition of consistency. More specifically, the grid can be considered as a

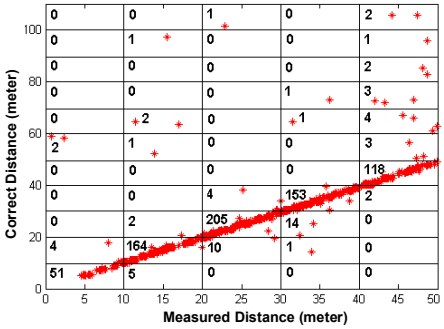

Figure 3. Superimposed grid with number of data points counted.

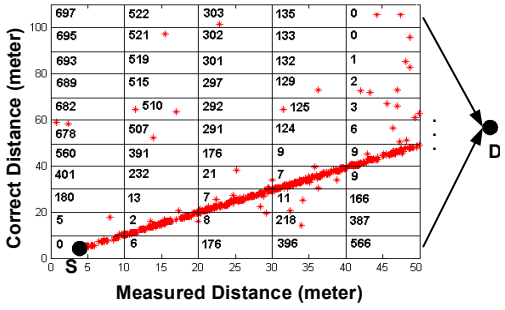

Figure 4. Inconsistency charge/cost and the modified Dijkstra's shorted path algorithm setup.

1. Let $L(1) = min(C[1, j]), j=1,\ldots,N$
2. for $i=2,\ldots,M$
   $L(i) = min(C[i, j]+L(i-1)), j=1,\ldots,N$
3. Trace the shortest path that lead to $L(M)$

Figure 5. Pseudocode of the modified Dijkstra's shortest path algorithm.

MxN matrix (M=5 and N=11 in the example). $C[i, j]$ denotes the number of data points within the grid cell $[i, j]$, where i=1,…,M and j=1,…,N. the inconsistency cost/charge of grid cell $[i, j]$ is defined as $\sum C[s, t]$ where $(s>i \wedge t<j) \vee (s<i \wedge t>j)$, s=1,…,M and t=1,…,N. Figure 4 shows the inconsistency cost of all gird cells with respect to all other grid cells.

Now we transform the problem instance of consistent and monotonic regression determination defined on a grid to the corresponding graphic theoretic instance where calculating the consistent and monotonic regression function is equivalent to finding the shortest path in the graph domain. The graph is constructed in the following way (Figure 4). The graph has $M \cdot N$+1 nodes where the $M \cdot N$ grid cells correspond to the $M \cdot N$ nodes in the graph. In addition, the graph has a node labeled as the destination node D. The node which corresponds with the grid cell [1, 1] is labeled as the source node S. The inconsistency cost/charge of each grid cell is the weight of each node. Moreover, $(M-1)$ $[1/2\ N(N+1)+(N-1)]$ directed edges are introduced in the following way. Each node that corresponds to the grid cell $[i, j]$ has outgoing edges to i) the node that corresponds to the grid cell $[i, j+1]$; ii) the nodes that correspond to grid cells $[i+1, t]$ where t=j,…,N. All N nodes belong to the last column have outgoing edge to the destination node. Once the graph is constructed, our goal is to find the path from the source to the destination which has the least accumulative weight. Note that this shortest path problem is equivalent to finding the most consistent regression function with respect to all data points since the assigned weights of the nodes are their inconsistency cost. Moreover, the monoticity of the regression function (i.e. the shortest path) is enforced by the way the edges are introduced. The problem of finding the shortest path in a graph can be solved using the Dijkstra's shortest path algorithm. Due to the special structure of the graph, even more efficient algorithm can be constructed in the following way. For the sake of simplicity, we describe the process in the grid domain. The idea is to traverse the grid cells in a certain order. Grid cells are traversed according to the column that they belong in, columns are visited in the left to right fashion and the grid cells within the same column are visited in the bottom up fashion. The goal is to calculate the accumulative minimum inconsistency cost among all paths from the source to the current column. Figure 5 states the pseudocode of this simplified Dijkstra's algorithm. The minimum inconsistency cost is first calculated along the first column (line 1). For each grid cell belong to column i, i=2,…,M, the accumulative minimum inconsistency cost up to the previous column is added to all the cells in the ith column without violating the monoticity principle. The process is speed up by only remembering the minimum accumulative inconsistency cost from the previous column $L(i-1)$ (line 2). The grid cell that has the minimum inconsistency cost in the last column is the second to last node on the shortest path of minimum weight (D is the last node on the path). The shortest path is determined by tracing back the grid cells that have leaded to the grid cell which has the least accumulative inconsistency cost in the last column (line 3).

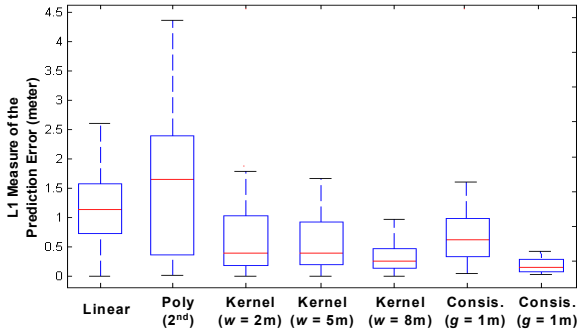

Figure 6. The prediction error of various error models.
Kernel: w denotes the sliding window size;
Consistency (Consis.): g denotes the grid size.

The runtime of the Dijkstra's algorithm is $O(V^2+E)=O(V^2)$ where V is the number of nodes in the graph and E is the number of edges. In our case, the runtime of the Dijkstra's algorithm becomes $O((MN)^2)$. Since our graph is sparse, the runtime of the Dijkstra's algorithm can be speed up using priority queue with binary heap to $O(ElgV)$, which is $O(M \cdot N^2 lg(M \cdot N))$ for our graph. The modified version of the algorithm presented in this section has runtime $O(M \cdot N)$. Therefore, we achieve a speed up of $O(Nlg(M \cdot N))$.

We evaluate the regression functions using the standard learn-and-test technique [7], where t% of the original data is used to conduct regression while the remaining portion of the data is used to evaluate the regression functions. In our study, t=70%. We randomly select 70% of the data points to construct the regression function; then we obtain the predicted values for the remaining data based on the regression function and compare the difference (L_1 norm) between the predicted value and the actual correct value for all the testing data. Figure 6 shows the boxplots of the L_1 prediction error cross four types of regression methods: linear regression, polynomial of second degree regression, kernel density estimation-based regression [7], and consistency-based regression. The top and the bottom line of each boxplot are the maximum and the minimum prediction error of the testing set respectively; the upper line of the box, the line in the box, and the bottom line of the box are the 75%, 50% (mean), and the 25% percentiles of the prediction error of the testing set respectively.

B. *Density Estimation*

In this section, we introduce the pair-wise consistency-based technique for the derivation of the density estimation function. We start by explaining the importance of the density estimation and restating the density estimation problem from pair-wise consistency point of view. Then we explain the consistency-based density estimation procedure. Finally, we discuss how the density estimation function are validated and evaluated.

Regression curve answers the question of what is the most likely value of the predicted variable for a given predicting

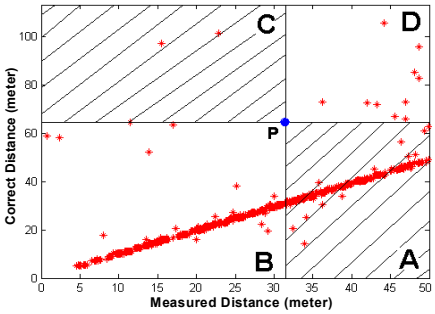

Figure 7. Positive Inconsistency calculation.

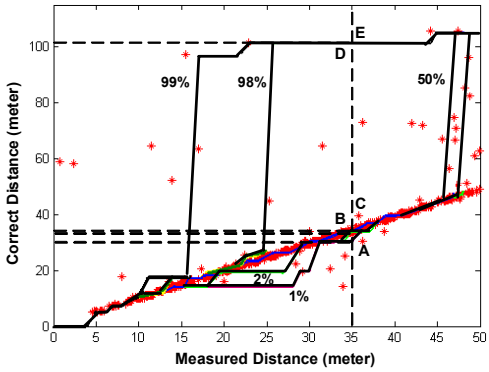

Figure 8. Five deferent regression lines.

variable. The goal of regression curve can be many-fold and includes the minimization of average and maximum error. Density estimation is a generalization of regression. It does not only indicate the most likely predicted variable given a predicting variable, but also provide the probability of any particular value of the predicted variable is observed for a given predicting variable. It is easy to see that density estimation is significantly difficult than regression and it often requires significantly many more pairs of predicting and predicted variables than regression itself. Usually density estimation function is presented graphically in 3-d space where x-axis corresponds to the predicting variable, y-axis corresponds to the predicted variable, and z-axis is the probability of pairs of x and y coordinates. Standard techniques for density estimation are based on histograms and smoothing the histograms using windows of different scope and shape. There are several conceptually different ways to enable the transition from pair-wise consistency-based regression curves to density estimation function. We will reside our attention only on one that results the best performance according to statistical and application tests. The overall approach has four phases: i) identification of subset of data points for regression; ii) consistency-based regression (Section 4.1); iii) cumulative density function (CDF) derivation; iv) probability density function (PDF) derivation.

We first identify the subset of points from the original set that have cardinality of *2C%*, where *C* is the percentage of the points that forms the lowest *C%* of the CDF. The identification of such points is based on the calculation of *positive inconsistency*. Positive inconsistency can be defined either in the original or the normalized form. In the original form, for the point with coordinates (x_i, y_i), positive inconsistency is equal to the difference in number of points (x_j, y_j) that have the property $(x_j>x_i \wedge y_j<y_i)$ and the number of points (x_k, y_k) that have the property $(x_k<x_i \wedge y_k>y_i)$. The concept of positive inconsistency is depicted in Figure 7 where the positive inconsistency of point *P* is the difference between the number of points in region *A* and region *C*. The stated definition is adequate when the points are approximately uniformly distributed with respect to the *x*-axis. However, when this is not the case, there is a need to compensate for the non-uniform distribution of points. For example, the normalization can be conducted by following Equation (4).

$$PI(P) = \frac{|A|}{|A|+|D|} - \frac{|C|}{|C|+|B|} \quad (4)$$

where |i| is the number of points in region *i*.

To summarize, positive inconsistency is a quantitative measure of how often a particular point is larger than expected with respect to all other points. Therefore, if the goal is to identify the values that form *C%* of the CDF, we have to identify *2C* points that have the lowest value (possibly negative) in terms of positive inconsistency. Note that once the least consistent points are identified, the relative ranking of the other points can be altered. Therefore, in principle, it is required to simultaneously select all *2C* points. We have experimented with various heuristic and probabilistic approaches for this task, and found that iteratively resorting is not required and it is sufficient to select points according to their initial ranking, at least for our sets of distance measurements. Out of all data sets (33 in total), this strategy is optimal in all but seven cases. Even in these seven cases, the ranking is minimally altered, and never more than two positions.

Once we identify the subset of cardinality of *2C* least consistent points, we can fit a regression curve based on the pair-wise consistency. We experimented with a variety of fits according to both statistical and numerical-based tests, pair-wise regression-based modeling performs the best. Figure 8 shows five regression curves of different percentages: 1%, 2%, 50%, 98% and 99%. A simple but crucial observation is that these regression curves correspond to the CDF value *C* at each measured distance. Figures 9 shows the CDF for a given measured distance 35m. The five points A, B, C, D and E in Figure 9 correspond to the five points on the CDF curve in Figure 8. By positioning a vertical line for each given predicting value (measured distance) and read off the predicted values (real distance) by following the regression curves of different *C* value, we can obtain the complete CDF curve. For each measured distance, its CDF can be consequently presented in a 2-d plot, where the x-axis is the predicted real distance and on the y-axis is the accumulative probability.

Once the CDF is available, it is easy to derive the PDF using either numerical or statistical techniques. Figure 10 shows the piece-wise linear estimation of the PDF. For the case of numerical technique, all is required is to take the difference between two consecutive values of the CDF function. In some case, it may be advantageous to employ statistical fitting in order to increase the smoothness of the PDF. From obtained PDFs and CDFs for various measured

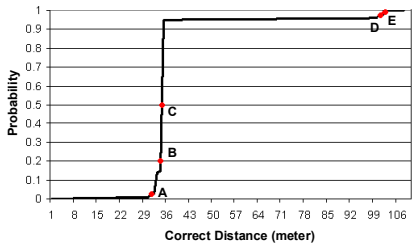

Figure 9. CDF of the predicted correct distance for the measured distance 35m.

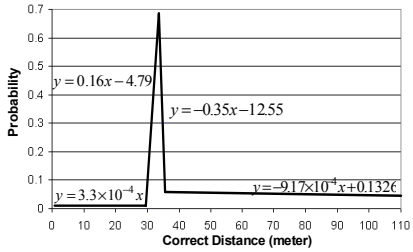

Figure 10. A piece-wise linear estimation of the PDF for the measured distance 35m.

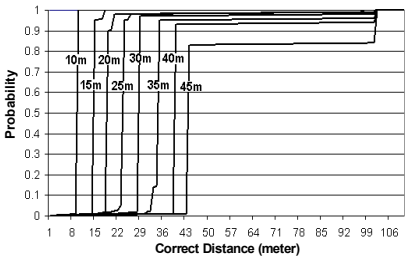

Figure 11. The CDFs of eight different measured distances.

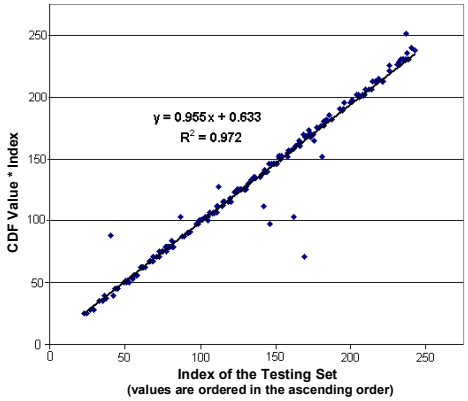

Figure 12. The CDF evaluation.

distances, we can easily construct 3-d probability estimation and accumulative probability estimation functions for distance measurements. For the sake of easy visualization, Figure 11 shows an example of 3-d CDF represented in a 2-d space for eight distance measurements.

The validation and evaluation of the derived CDF and PDF are conducted also using the learn-and-test technique. We use 70% of the original data as the training set to derive the CDF, the remaining 30% is testing set and we did 200 resamplings. The key idea is to map each data point in the testing set to its corresponding CDF value, which is derived using the learning data set. After each resampling, we plot the sorted (in ascending order) testing sets where the x-coordinate indicates its ranking normalized against the cardinality of the testing data set, and the y-coordinate shows the product of its CDF value and its ranking (Figure 12). Note that if the CDF derived based on the training data is a good representation of the testing set, all points would reside on the line $y = x$. By examining the slope (0.955) and the residual (0.972) of the least linear squares fit, we conclude that the CDF (therefore the PDF) obtained using the regression techniques described in Section 4.1 is indeed an accurate representation of the distance measurement errors.

Once the PDF is available, we modify the objective function in such a way that the probabilities of certain error values occurring are maximized. This is based on the standard assumption that errors are independent. The function M (in Equation (2)) no longer solely depends on the single variable of ε_{ij}, but also the measurement d_{ij} itself. More specifically, the objective function F has the form expressed in Equation (5) and is subject to maximization, where P_{ij} is the probability that error ε_{ij} is detected when the estimated distance between sensors i and j is d_{ij}.

$$F = \prod P_{ij}, \text{ where } P_{i,t} = M(\varepsilon_{i,t}, d_{i,t}) \quad (5)$$

for pairs of nodes i&j that have measured distance d_{ij}

In the actual implementation of the nonlinear function minimization, instead of maximizing the product of the probabilities, we take the logarithm of each probability and maximize the summation of logarithms.

V. Experimental Results

In this section, we experimentally evaluate the three consistency-based LD algorithms: GPS-based and GPS-less off-line LD based on consistency error models; the GPS-based on-line LD. In GPS-less LD, we first solve the instance without using any beacon information (obtain relative locations); then map the relative locations to the absolute positions using the available beacon information. The executions cross all four scenarios are done on a Pentium III 1200MHz processor. We conduct analysis of the LD algorithms in terms of the average connectivity, and the scalability in terms of network size, dimension and different types of measurement errors. In addition, we also present the results for 10 other randomly selected data sets. Finally, we compare the relative performance of the LD algorithms with a sample of previously published algorithms. All experiments are conducted based on the data produced by the deployed sensor networks.

A good way to evaluate the overall effectiveness of both the objective function and the LD algorithm is to compare the input error (the distance measurements errors) and the resultant location errors. Figures 13(a) and 13(b) present the boxplots of the distance measurement errors. The median and

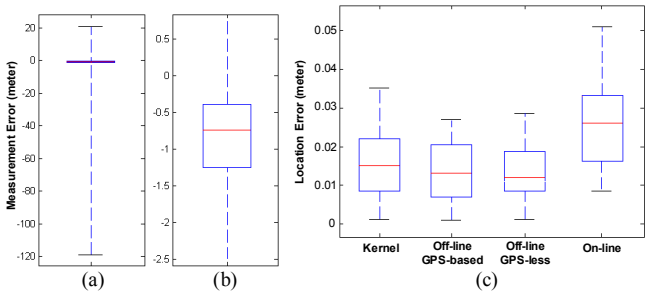

Figure 13(a) The measurement error (measured – real) boxplot. 13(b). The measurement error boxplot zoom view. 16(c). The boxplots of the location error comparison.

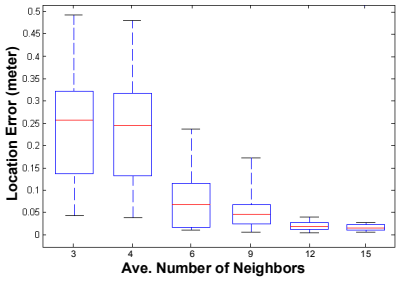

Figure 14. The location error boxplots given different average connectivity for off-line GPS-less LD.

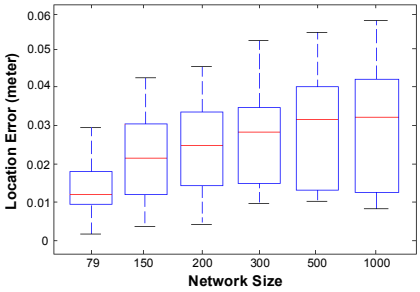

Figure 15. The scalability study – location error boxplots given different network sizes.

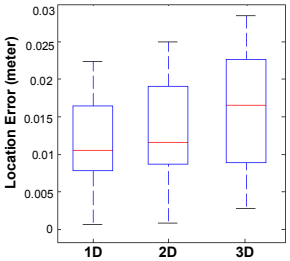

Figure 16. The location error boxplots for different dimensions.

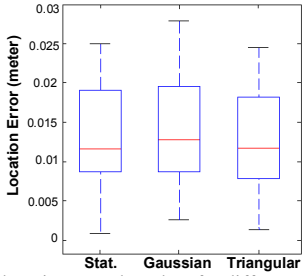

Figure 17. The location error boxplots for different types of errors in measurements.

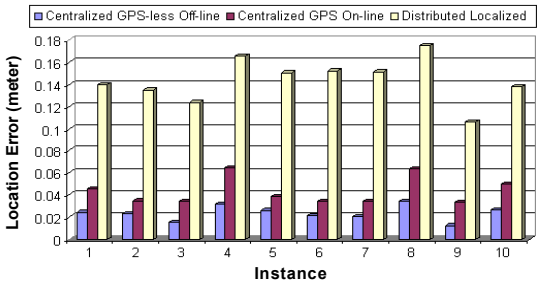

Figure 18. The median location error comparison of the centralized off-line LD, the centralized on-line LD, and the localized LD across 10 independent data sets.

average of the measurement error are 6.73m and 0.74m respectively. Figure 13(c) presents the boxplots of the location errors for five algorithms: the centralized off-line algorithm with error model constructed using the kernel density estimation technique as the optimization objective [21]; the centralized GPS-based and GPS-less off-line algorithm with consistency-based error models as the optimization objectives; the centralized on-line algorithm; and finally the localized algorithm. We conclude from the plot that without considering the beacons (GPS-less) yields better median location error than in the case of when beacons are available. Our interpretation for this is that the optimization has more degrees of freedom to alter each node's positions around in order to improve the objective function as suppose to when the beacons' positions are fixed. We compare the relative performance with a recent state-of-the-art literature [1] in terms of the ratio of the resultant location error and the input error (random noise). The authors introduced random noise which follows the Gaussian distribution with mean 0 and standard deviation 1cm, 5cm and 10cm. The resultant mean-square errors are 4.43cm, 14.39cm and 16.22cm respectively (e.g. the mean location errors are then 2.1cm, 3.8cm and 4.02cm respectively). Therefore, the corresponding ratio between the location error and the input error are 210%, 76% and 40.2% respectively. In our study, we consider the mean location error of the four algorithms and then normalize them against the mean input error (0.74m), the corresponding ratios are 1.8% (off-line GPS-based), 1.76% (off-line GPS-less), 3.7% (on-line GPS-based).

It is widely assumed that a high degree of connectivity of nodes resolves to smaller location errors. Figure 14 show the boxplots of the location error distribution given different average number of LD neighbors for the off-line GPS-less. We see that while it is important to have more than minimally required three neighbors, once the number of neighbors per node is more than 10, one can expect very little further improvement. More importantly, the quality of the neighboring measurements matters much more than the sheer number of neighbors.

We have developed an integer linear programming (ILP)-based instance generator, which creates instances with random node placements while following a specified measurement error distribution [21]. The scalability analysis is conducted on the networks created using the ILP-based instance generator with the same error distribution as in the original instance (Figure 11). We use the centralized off-line GPS-less LD approach for this study.

From Figure 15, we observe that initially the median location error increases by more than a factor of 2 when the network size doubles (79 nodes to 150 nodes). However, the increase diminishes with any further size increase. In addition, we observe that the location error distribution expands to a wider range as the network size grows. This is an expected consequence of the presence of large number of nodes. Our interpretation of this phenomenon is that some nodes have higher probability of getting 'lucky' and vise versa when the

network size expands. It is interesting to note that no instances larger than 300 nodes are solved well using the centralized execution. Obviously the limit that can be addressed by the optimization software is reached. The instances larger than this critical point – 300 nodes, are solved by grouping 200 nodes consecutively and invoking the optimization in a distributed fashion.

In addition to network size, we also analyze the scalability in terms of dimensions and different types of errors in measurements. Figure 16 shows the location error boxplots when the localization is conducted in 1-d, 2-d and 3-d space. It is interesting to note that in 3-d, the medium and the 75% percentile of the location error increased by almost 50% while the other percentiles have smaller fluctuations. In Figure 17, we compare the performance on three sets of measurements that follow different types of error distribution. *Stat.* is the set of measurements we have obtained from the deployed networks which has error distribution shown in Figure 11 (the same GPS-less off-line boxplot as in Figure 13(c)). The other two sets of measurements are generated in simulation. On top of the real distances, random noise that i) follows the Gaussian distribution (μ=0, σ=0.5m) and ii) has triangular shape (h=0.5m, b=±0.5m) are imposed. The mean location errors are within 15% of each other for all three sets of measurements. This finding supports that the consistency-based error model (as the optimization target) is effective regardless of the types of error distribution.

Furthermore, we examined the consistency of performance on all of the 33 data sets. For the sake of easy visualization, Figure 18 shows the results for 10 randomly selected instances where the number of neighbors is on average six per node. Centralized GPS-less off-line, centralized GPS-based on-line, and the localized algorithms are evaluated.

VI. CONCLUSION

We have developed a new localization approach that is solely based on the concept of consistency. If the localization is conducted off-line, statistical error models with specified properties are constructed using non-parametric statistical methods and used to guide the optimization mechanism for localization. If the localization is conducted on-line, no a priori knowledge about the error distribution is necessary; the optimization objective is the consistency between the measurements and the solutions provided by the optimization mechanism. The approach is evaluated using data produced from deployed networks. We also compared the performance with several other state-of-the-art localization methods.

REFERENCES

[1] Moore, D., Leonard, J., Rus, D., and Teller, S. Robust distributed network localization with noisy range measurements. *2nd International Conference on Embedded Networked Sensor Systems*. (Nov. 2004), 50–61.

[2] Sayed, A., Tarighat, A., and Khajehnouri, N. Network-based wireless location. *IEEE Signal Processing Magazine*. 22(4), (July 2005), 24–40.

[3] Ward, A., Jones, A., Hopper, A. A new location technique for the active office. *IEEE Personal Communications*. 4(5), (1997), 42–47.

[4] Bahl, P., Padmanabhan. V.N. RADAR: an in-building RF-based user location and tracking system. *IEEE INFOCOM*. 2, (2000), 775–84.

[5] Hightower, J., Want, R., Borriello, G. *SpotON: An Indoor 3d Location Sensing Technology Based on RF Signal Strength*. CSE, University of Washington, WA, (2000).

[6] Savarese, C., Rabaey, J., Langendoen, K. Robust positioning algorithms for distributed ad-hoc wireless sensor networks. *USENIX Technical Annual Conference*. (2002), 317–327.

[7] Hastie, T., Tibshirani, R., and Friedman, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction,*. Springer-Verlag. NY, 2001.

[8] Merrill, W., Girod, L., Elson, J., Sohrabi, K., Newberg, F., and Kaiser, W. Autonomous position location in distributed, embedded, wireless systems. *IEEE CAS Workshop on Wireless Communications and Networking*. (2002).

[9] Thisted, R., *Elements of Statistical Computing: Numerical Computation*. CRC Press, NY, (1988).

[10] Merrill, W., Newberg, F., Girod, L., and Sohrabi, K. Battlefield ad-hoc LANs: a distributed processing perspective. *GOMACTech*, (2004).

[11] Patwari, N., Ash, J.N., Kyperountas, S., Hero, A.O., Moses, R.L., and Correal, N.S. Locating the nodes. *IEEE Signal Processing Magazine*. 22(4), (July 2005), 54–69.

[12] Feng, J., Koushanfar, F., and Potkonjak, M. Localized algorithms for sensor networks. *Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems*. CRC Press, (2004).

[13] Savvides, A., Han, C., and Strivastava, M.B. Dynamic fine-grained localization in ad-hoc networks of sensors. MOBICOM, (2001), 166–179.

[14] Sheng, X., and Hu, Y.H. Energy based acoustic source localization. *ACM/IEEE International Symposium on Information Processing in Sensor Networks*, (2003), pp. 285–300.

[15] Hu, L., and Evans, D. Localization for mobile sensor networks. *9th annual international conference on Mobile computing and networking*. (2003), 45–57.

[16] Niculescu, D., and Nath, B. Error characteristics of ad hoc positioning systems (APS). 5th *ACM International Symposium on Mobile Ad Hoc Networking and Computing*, (2004), 20–30.

[17] Galstyan, A., Krishnamachari, B., Lerman, K., and Pattem, S. Distributed online localization in sensor networks using a moving target. *ACM/IEEE International Symposium on Information Processing in Sensor Networks*. (2004), 61–70.

[18] Gustafsson, F., and Gunnarsson, F. Mobile positioning using wireless networks. *IEEE Signal Processing Magazine*. 22(4), (July 2005), 41–53.

[19] He, T., Huang, C., Blum, B.M., Stankovic, J.A., and Abdelzaher, T. Range-free localization schemes for large scale sensor networks. *9th annual international conference on Mobile computing and networking*, (2003), 81–95.

[20] Shang, Y., Ruml, W., Zhang, Y., and Fromherz, M.P.J. Localization from mere connectivity. *4th ACM international symposium on Mobile ad hoc networking & computing*, (2003), 201–212.

[21] Feng, J., and Potkonjak, M. Location Discovery using Data-Driven Statistical Error Modeling. To appear in *InfoCom'06*, Barcelona, (2006).