2005 IEEE 6th Workshop on Signal Processing Advances in Wireless Communications

DISTRIBUTED GAUSS-NEWTON METHOD FOR NODE LOCLAIZATION IN WIRELESS SENSOR NETWORKS

Bing Hwa Cheng, Ralph E. Hudson, Flavio Lorenzelli,
Lieven Vandenberghe, Kung Yao,
UCLA

ABSTRACT

We present distributed algorithms for sensor localization based on the Gauss-Newton method. Each sensor updates its estimated location by computing the Gauss-Newton step for a local cost function and choosing a proper step length. Then it transmits the updated estimate to all the neighboring sensors. The proposed algorithms provide non-increasing values of a global cost function. It is shown in the paper that the algorithms have computational complexity of O(n) per iteration and a reduced communication cost over centralized algorithms.

1. INTRODUCTION

The use of sensor networks for monitoring has been increasing extensively. The location of each sensor must be determined before performing any useful monitoring. For large number of nodes in the network, it is not possible to have GPS capability or precise calibration for each sensor. Several techniques have been proposed using centralized algorithms. [1] and [2] use convex optimization to estimate the locations of sensor nodes given range information, i.e., pair-wise distance measurements between sensor nodes. In [6], an iterative non-linear least-squares method is proposed to estimate sensor locations when no sensor with known location (anchor) is available. However, as the number of sensors grows, it is not desirable to have a centralized algorithm because of the computational complexity and communication cost. Several distributed methods have been proposed. In [9], ad-hoc node localization method is proposed. In this approach each sensor counts the number of hops to 3 nearest anchors and computes average hop distance. With this information each sensor can perform multilateration to obtain the position estimate. A distributed refinement to [9] using triangulation is presented in [14]. The N-hop multilateration method in [10] provides distributed estimation of the sensor location by dividing the network into several subtrees such that the solution in each subtree is uniquely determined. In [11], a similar approach is proposed. A dense network is divided into several clusters. Each cluster estimates its own sensors locations using semidefinite programming (SDP) in [1]. If the sensor is in multiple clusters, the algorithm simply chooses the best estimation among those clusters. The nonparametric belief propagation (NBP) method in [7] estimates the locations of sensors in a distributed fashion. Each sensor sends different messages to its neighboring sensors, where each massage consists of random samples and marginal estimates of the corresponding neighboring sensor. In [13], distributed weighted multi-dimensional scaling (dwMDS) is presented when a routing loop in the network is available.
We present distributed algorithms in which each sensor has noisy measurements of the distance to its neighboring sensors and anchors (sensors with known locations). The measurements can be obtained using RSSI (Received Signal Strength Indicator) or ToA (time-of-arrival) described in [12]. Based on the measurements of the distance and estimated locations of the neighboring sensors, we update the estimated location by finding a descent step using Gauss-Newton method on the local cost function and choosing a proper step length. Then we broadcast the updated estimation to the neighboring sensors. Based on how we update the estimated locations, we present two distributed algorithms: sequential and parallel algorithms. In the sequential algorithm, each sensor computes a descent step, finds a step length and then updates its estimated location sequentially. In the parallel algorithm, all sensors first compute their descent steps simultaneously. Then each sensor finds step length and updates its location estimate sequentially. Since computing a step requires more processing time than finding a step length, the parallel algorithm will be faster than the sequential algorithm. It is shown in [15] that both algorithms provide non-increasing values of a global cost function.

2. DISTRIBUTED ALGORITHMS

Supposed we have m sensors with known locations (anchors) $a_k \in R^2$, $k=1,\dots,m$, and n sensors $x_i \in R^2$, $i=1,\dots,n$ whose locations are unknown and need to be determined. Each sensor has distance measurements to all the neighboring sensors and anchors. For example, the i-th sensor has Euclidean distance measurement d_{ij} between x_i, x_j and d_{ik} between x_i, a_k if the distance between

x_i, x_j and between x_i, a_k is less than the radio range R. We assume there is a communication link established between a pair of sensors if the distance is less than the radio range.

Given all the range information d_{ij} and d_{ik}, the localization problem [1] is to find x_is such that

$$r_{ij}(\mathbf{x}) = \|x_i - x_j\|^2 - d_{ij}^2 \approx 0, \text{ if } \|x_i - x_j\| \le R,$$
$$s_{ik}(\mathbf{x}) = \|x_i - a_k\|^2 - d_{ik}^2 \approx 0, \text{ if } \|x_i - a_k\| \le R, \quad (1)$$

where $\mathbf{x} = \left[(x_1)^T \quad (x_2)^T \quad \cdots \quad (x_n)^T\right]^T$.

This problem can be formulated as a non-linear least-squares problem, i.e.,

$$\min_{\mathbf{x}} F(\mathbf{x}) = \min_{\mathbf{x}} \left\{ \sum_{i,j} |r_{ij}(\mathbf{x})|^2 + \sum_{i,k} |s_{ik}(\mathbf{x})|^2 \right\}. \quad (2)$$

A centralized Gauss-Newton method for solving (2) is given in [15].

2.1. Sequential Algorithm

We first define the local cost function of node i as

$$F_i(x_i, x_{j|j\to i}) = \sum_{j\to i} \left| \|x_i - x_j\|^2 - d_{ij}^2 \right|^2 + \sum_{k\to i} \left| \|x_i - a_k\|^2 - d_{ik}^2 \right|^2, \quad (3)$$

where $j \to i$ and $k \to i$ represent all the sensors and anchors that are neighbors of node i, respectively. Assuming all the neighboring sensors before sensor i have been updated at iteration $t+1$, that is, $x^{(t+1)}_{j|j\to i, j<i}$ are available to sensor i. For the rest of the neighboring sensors, $x^{(t)}_{j|j\to i, j>i}$ are also available from the previous iteration t. Given $x^{(t+1)}_{j|j\to i, j<i}$ and $x^{(t)}_{j|j\to i, j>i}$, we can formulate the problem of minimizing the local cost function in (3) as a non-linear least-squares problem with variable x_i, i.e.,

$$\min_{x_i} F_i(x_i, x^{(t+1)}_{j|j\to i, j<i}, x^{(t)}_{j|j\to i, j>i}). \quad (4)$$

Given the estimated location of sensor i at iteration t, $x_i^{(t)}$, we can approximate (4) as a linear least-squares problem using the Gauss-Newton method, i.e.,

$$\min_{p_i} \left\{ \sum_{j\to i, j<i} \left| r_{ij}(x_i^{(t)}, x_j^{(t+1)}) - \nabla r_{ij}(x_i^{(t)}, x_j^{(t+1)})^T p_i \right|^2 + \sum_{j\to i, j>i} \left| r_{ij}(x_i^{(t)}, x_j^{(t)}) - \nabla r_{ij}(x_i^{(t)}, x_j^{(t)})^T p_i \right|^2 + \sum_{k\to i} \left| s_{ik}(x_i^{(t)}) - \nabla s_{ik}(x_i^{(t)})^T p_i \right|^2 \right\}. \quad (5)$$

Let p_i^{Seq} be the solution to (5), the update of sensor i is given by $x_i^{(t+1)} = x_i^{(t)} - \alpha_i p_i^{Seq}$, where α_i is a proper step length. Then sensor i transmits the updated estimation $x_i^{(t+1)}$ to all the neighboring sensors and sensor $i+1$ starts to compute the step and find a step length. The $(t+1)$-st iteration ends when all sensors have updated their estimations.

2.2. Parallel Algorithm

One disadvantage of the sequential algorithm is that all steps have to be computed in sequentially. We present a parallel algorithm based on the Jacobi method [3] which allows the computation done in a parallel fashion. Given all the estimated locations of the neighboring sensors of node i at iteration t, that is, $x^{(t)}_{j|j\to i}$, we formulate the problem of minimizing the local cost function (3) as a non-linear least-squares problem with variable x_i, i.e.,

$$\min_{x_i} F_i(x_i, x^{(t)}_{j|j\to i}). \quad (6)$$

The approximation to a linear least-squares problem is given by

$$\min_{p_i} \left\{ \sum_{j\to i} \left| r_{ij}(x_i^{(t)}, x_j^{(t)}) - \nabla r_{ij}(x_i^{(t)}, x_j^{(t)})^T p_i \right|^2 + \sum_{k\to i} \left| s_{ik}(x_i^{(t)}) - \nabla s_{ik}(x_i^{(t)})^T p_i \right|^2. \right\} \quad (7)$$

Let p_i^{Para} be the solution to (7), the update of sensor i is given by $x_i^{(t+1)} = x_i^{(t)} - \alpha_i p_i^{Para}$, where α_i is a proper step length. Note that all the steps $p_i^{Para} \in R^2, i = 1,\ldots,n$ can now be computed simultaneously. In general, the parallel algorithm requires slightly more iterations than the sequential algorithm for the same accuracy. Thus the total computational complexity is slightly increased. But the parallel algorithm has much faster processing time since all the computations are done simultaneously.

3. STEP LENGTH SELECTION

In the sequential algorithm, the step length selection can be done using general techniques such as the backtracking line search [5] sequentially on each sensor, that is, find a $\alpha_i > 0$ such that

$$F_i(x_i^{(t)} - \alpha_i p_i^{Seq}, x_{j|j\to i, j<i}^{(t+1)}, x_{j|j\to i, j>i}^{(t)}) \le F_i(x_i^{(t)}, x_{j|j\to i, j<i}^{(t+1)}, x_{j|j\to i, j>i}^{(t)}). \tag{8}$$

In the parallel algorithm, in order to guarantee non-increasing values of global cost function, the step length selection also needs to be done sequentially, that is, at sensor i, find a $\alpha_i > 0$ such that

$$F_i(x_i^{(t)} - \alpha_i p_i^{Para}, x_{j|j\to i, j<i}^{(t+1)}, x_{j|j\to i, j>i}^{(t)}) \le F_i(x_i^{(t)}, x_{j|j\to i, j<i}^{(t+1)}, x_{j|j\to i, j>i}^{(t)}). \tag{9}$$

It is shown in [15] that both sequential and parallel algorithms provide non-increasing values of the global cost function $F(\mathbf{x})$. Both algorithms have computational complexity of $O(n)$ per iteration and only local information (estimated locations from the neighboring sensors) is required. Note that it is possible in the parallel algorithm that some sensors will find zero step length, i.e., $\alpha_i = 0$ since p_i^{Para} is a descent step with respect to the local cost function $F_i(x_i, x_{j|j\to i}^{(t)})$, not $F_i(x_i, x_{j|j\to i, j<i}^{(t+1)}, x_{j|j\to i, j>i}^{(t)})$. But from practical experience, only few sensors in the network will find zero step length. Thus the overall value of cost function is still decreasing. We summarize sequential and parallel algorithms Figure 1.

We assume there is a central node that sends out activation messages sequentially to all sensors. After each sensor receives the activation message, it starts processing the information. In the sequential algorithm, it computes the step, finds the step length, updates the estimation and then broadcasts the estimation to its neighboring sensors. In the parallel algorithm, since the step is already computed at the beginning of the iteration, it only needs to find the step length, update the estimation and broadcast the estimation to its neighboring sensors. Denote T_1 and T_2 (seconds) the processing time at each sensor for sequential and parallel algorithms, respectively. Then an activation message is sent every T_1 seconds for the sequential algorithm, and every T_2 seconds for the parallel algorithm. Thus sensors do not need to communicate back to the central node. Also if one sensor crashes, the rest of the sensors can still continue the updating process. Note that adjacent sensors, x_i, x_{i+1}, are not necessarily neighbors to each other. Thus no pre-specified routing loop is required.

Each sensor can determine its own stopping criterion by comparing the magnitude of the step to a given threshold, i.e., stop computing step and updating if $\|p_i\| < \varepsilon$ where ε is a pre-determined value.

```
Sequential algorithm
Given starting point $\mathbf{x}^{(t)}$
for i=1:1:n
    • Compute $p_i^{Seq}$ in (5).
    • Find step length $\alpha_i$ such that (8) holds.
    • $x_i^{(t+1)} = x_i^{(t)} - \alpha_i p_i^{Seq}$
    • Transmit $x_i^{(t+1)}$ to all the neighboring sensors.
end
t=t+1
```

```
Parallel algorithm
Given starting point $\mathbf{x}^{(t)}$
Compute $\mathbf{p}^{Para}$ in (7).
for i=1:1:n
    • Find step length $\alpha_i$ such that (9) holds.
    • $x_i^{(t+1)} = x_i^{(t)} - \alpha_i p_i^{Para}$.
    • Transmit $x_i^{(t+1)}$ to all the neighboring sensors.
end
t=t+1
```

Figure 1. Sequential and parallel algorithms

4. SIMULATION RESULTS

In this section we present some simulation results. Comparisons of the complexity among different algorithms are also given. We consider a network with n and $n/10$ randomly chosen sensors and anchors, respectively. If the distance between two sensors or sensor and anchor is less than the radio range R, a noisy measurement of the distance is given by [1]

$$\hat{d}_{ij} = d_{ij}(1 + randn \times \text{noise factor}),$$

(10)

where noise factor is a given number related to the accuracy of the distance measurement and *randn* is a standard normal random variable with zero mean and unit variance.

First, we show an example of 100 sensors and 10 anchors placed randomly in a 1.6×1.6 region where the radio range is 0.35 and the noise factor is 0.1. The starting points are chosen as random perturbations from the true sensor locations, where the perturbation is given by uniformly distributed random variable in [*-0.1 0.1*]. The estimation results of the centralized, sequential and parallel algorithms are given in Figure 2, represented by "x". The true sensor locations are given by the "o". Anchors are given by "v". It can be seen from Figure 2 that distributed algorithms described in section 2 converge to almost the same results as the centralized Gauss-Newton algorithm. Figure 3 shows comparisons of the convergence behavior of the parallel algorithm, the distributed triangulation (or one-hop multilateration) method in [10] and [14] over 5 random realizations.

Then, we increase the size of the network n while keeping the density of both sensors and anchors unchanged, that is, n sensors and $n/10$ anchors located randomly in a $1.6\sqrt{\frac{n}{100}}\times 1.6\sqrt{\frac{n}{100}}$ region. Figure 4 shows average numbers of iteration over 20 realizations for both sequential and parallel algorithms to reach a given accuracy (within ± 0.01 from the true location). The radio range is 0.35 and noise factor is set to zero in all cases. It can be seen that the sequential algorithm requires slightly fewer iteration. But the total processing time in Figure 5 is much larger for the sequential algorithm. This is because the computation of the steps p_i^{Seq} s in the sequential algorithm must be computed sequentially, while p_i^{Para} in the parallel algorithm can be computed simultaneously (for simplicity, we ignore the processing time for finding step length). Finally, we present some empirical results on the communication energy cost. We use the model given by [8] that the propagation loss is proportional to the fourth power of distance. For centralized algorithms, each sensor only needs to transmit to the central processor once. However, as the size of the network increases or equivalently, the region increases, each sensor will need more energy to transmit the information to a central processor (we assume the central processor lies at location $(0,0)$). In distributed algorithms, each sensor only communicates to the neighboring sensors. Thus the total energy cost depends linearly on the total number of sensors and number of iteration required. Figure 6 shows the average energy consumption over 20 realizations of different algorithms. It can be seen when the size of network is large, distributed algorithms have reduced energy consumption.

5. CONCLUSIONS

We have presented distributed algorithms for localization in wireless sensor networks based on Gauss-Newton method. Both algorithms provide non-increasing values of the global cost function. The proposed methods have linear computational complexity per iteration and reduced communication energy consumption over centralized algorithms.

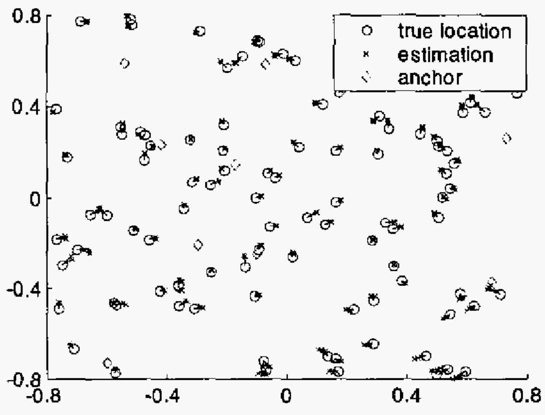

Figure 2. Estimation results of the centralized, sequential and parallel algorithms

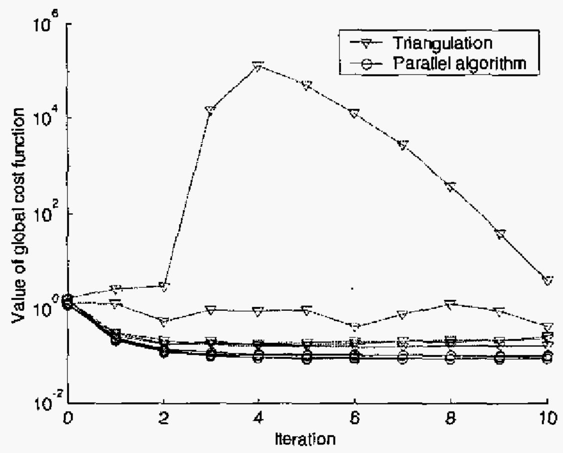

Figure 3. Estimation error versus iteration for different algorithms over 5 random realizations

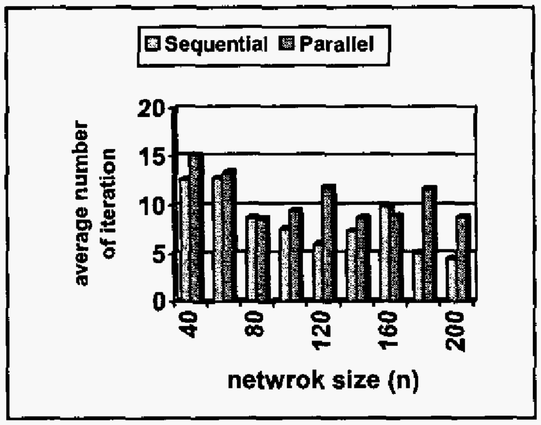

Figure 4. Average number of iteration

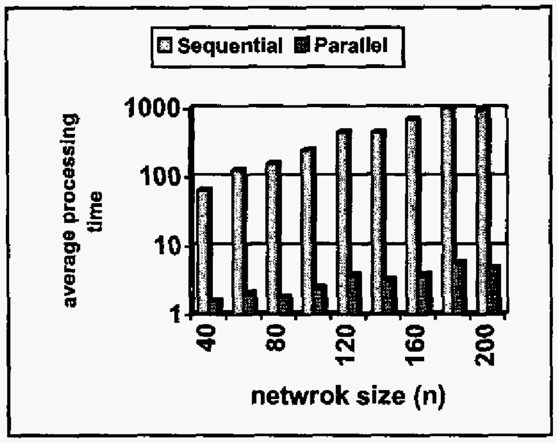

Figure 5. Average processing time (in log scale)

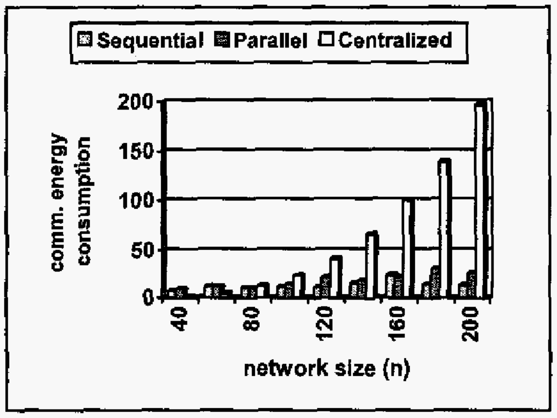

Figure 6. Average energy consumption due to communication

7. REFERNCES

[1] P. Biswas, Y. Ye, "*Semidefinite Programming for Ad Hoc Wireless Sensor Network Localization*", IPSN, Pages 46-54, April 2004.

[2] L. Doherty, L. E. Ghaoui, and S. J. Poster. "*Convex Position Estimation in Wireless Sensor Networks*" IEEE *Infocom*, v.3, pages 1655-1663, April 2001.

[3] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation*. Englewood Cliffs, 1989.

[4] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Non-linear Equations in Several Variables*, Academic Press, New York and London, 1932.

[5] S. Boyd, L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.

[6] R. Moses, D. Krishnamurth and R. Patterson, "*Self-localization for Wireless Networks*", Eurasip Journal on Applied Signal Processing, pages 348-358, 2003.

[7] A. Ihler, J. Fisher III, R. Moses and A. Willsky, "*Nonparametric Belief propagation for Self-Calibration in Sensor Networks*", IPSN, pages 225-233, April 2004.

[8] G. Pottie and W. Kaiser, "Wire*less Integrated Network Sensors*", Communications of the ACM, 43(5): 51-58, May 2000.

[9] D. Nicolescu, B. Nath, "*Ad-hoc Positioning System*", Proceedings of IEEE GLOBECOM, vol. 5, pages 2926-2931, Nov. 2001.

[10] A. Savvides, H. Park, M. B. Srivastava, "*The Bits and Flops of the N-hop Multilateration Primitive For Node Localization Problems*", ACM Mobile Networks and Applications, 8(4), 443-451 (2003).

[11] P. Biswas, Y. Ye, "*A Distributed Method for Solving Semidefinite Programs Arising From Ad-hoc Wireless Sensor Network Localization*", Technical report, Dept. of Management Science and Engineering, Stanford University, Oct. 2003.

[12] A. Savvides, S. Han, M. B. Srivastava, "*Dynamic fine-grained localization in Ad-Hoc networks of sensors*", Proceedings of the Seventh ACM Annual International Conference on Mobile Computing and Networking (MobiCom), July 2001.

[13] J. A. Costa, N. Patwari and A. O. Hero III, "*Achieving High-Accuracy Distributed Localization in Sensor Network*", ICASSP 2005.

[14] C. Savarese, J. Rabay and K. Langendoen, "*Robust Positioning Algorithms for Distributed Ad-Hoc Wireless Sensor Networks*", USENIX Technical Annual Conference, June 2002.

[15] B. H. Cheng, "*Distributed Gauss-Newton Method for Node Localization in Wireless Sensor Networks*", Technical Reports, 2005.