Mobile Networks and Applications 8, 443–451, 2003

The *n*-Hop Multilateration Primitive for Node Localization Problems

ANDREAS SAVVIDES, HEEMIN PARK and MANI B. SRIVASTAVA
Networked and Embedded Systems Lab, Electrical Engineering Department, University of California, Los Angeles, 6731-H Boelter Hall, Box 951594, Los Angeles, CA 90095-1594, USA

Abstract. The recent advances in MEMS, embedded systems and wireless communication technologies are making the realization and deployment of networked wireless microsensors a tangible task. In this paper we study node localization, a component technology that would enhance the effectiveness and capabilities of this new class of networks. The *n*-hop multilateration primitive presented here, enables ad-hoc deployed sensor nodes to accurately estimate their locations by using known beacon locations that are several hops away and distance measurements to neighboring nodes. To prevent error accumulation in the network, node locations are computed by setting up and solving a global non-linear optimization problem. The solution is presented in two computation models, centralized and a fully distributed approximation of the centralized model. Our simulation results show that using the fully distributed model, resource constrained sensor nodes can collectively solve a large non-linear optimization problem that none of the nodes can solve individually. This approach results in significant savings in computation and communication, that allows fine-grained localization to run on a low cost sensor node we have developed.

Keywords: ad-hoc localization, distributed localization, sensor networks

1. Introduction

Precise knowledge of node location in ad-hoc deployed microsensor networks yields a wide variety of profound advantages. Knowledge of location can be used to report the geographical origin of events, to assist in target tracking, geographic aware routing [11], to administer the sensor network and evaluate its coverage [12]. These together with security and smart environment applications such as the Smart Kindergarten [20] are only a few of the applications where location aware nodes are required.

In many situations, wireless sensor nodes are expected to be deployed in an ad-hoc fashion (i.e., air-dropped over an area). With ad-hoc deployment however, one cannot accurately predict or plan a-priori the location of each sensor. Based on this and the fact that direct line-of-sight with beacons is not always feasible, we seek to develop an algorithm that can perform precise localization of sensor nodes with indirect line-of-sight by utilizing location information and distance measurements over multiple hops. To achieve this goal, nodes use their ranging sensors to measure distances to their neighbors and share their measurement and location information for a small subset of similar nodes that are already aware of their locations, the beacon nodes. This sharing allows nodes to collectively estimate their locations.

In this paper we present the *n-hop multilateration primitive* that we also refer to as *collaborative multilateration*. Collaborative multilateration consists of a set of mechanisms that enables nodes found several hops away from beacon nodes to collaborate with each other and estimate their locations with high accuracy. This multihop operation waives the line-of-sight to beacons requirement making fine-grained localization possible in the presence of very few beacon nodes. Position estimates are obtained by setting up a global non-linear optimization problem and solving it using iterative least squares. Collaborative multilateration is presented in two computation models, centralized and distributed. These can be used in a wide variety of network setups from fully centralized where all the computation takes place at a base station, to locally centralized (i.e., computation takes place at a set of cluster heads) to fully distributed where computation takes place at every node.

The fully distributed computation model presented here is an approximation of its centralized counterpart and has several properties favorable to sensor networks. It offers a significant reduction in computation requirements thus allowing the execution of collaborative multilateration on resource constrained sensor nodes such as the Medusa MK-2 node described in this paper. Using this mechanism, resource-constrained nodes can collaborate with each other to jointly estimate their locations, a task that none of the nodes can perform individually because of their computation and memory limitations. The use of a fully distributed computation model is also tolerant to node failures, and distributes the communication cost evenly across the sensor nodes and does not require any additional supporting mechanisms such as leader election and multihop routing that would be required for a fully centralized implementation.

The algorithms presented in this paper are validated on a combined ns-2 and MATLAB simulation testbed, using the measured parameters of our experimental sensor nodes. The remainder of this paper is organized as follows. Next section briefly describes the related work. Section 3 provides some preliminary information and overview of our approach.

Section 4 describes the initial setup configuration for collaborative multilateration. Section 5 explains the centralized and distributed computation models. Our simulation results and a brief comparison to the corresponding Cramér–Rao bound results are presented in section 6 and section 7 concludes the paper.

2. Related work

Node localization has been the topic of active research and many systems have made their appearance in the past few years. A detailed survey of such systems is provided by Hightower and Boriello in [7]. Despite these efforts, very few systems are actually ad-hoc. Even fewer methods are have been proposed that offer a fully distributed operation. Doherty's [3] convex position estimation approach for instance describes a method for localizing ad-hoc nodes based only on connectivity. This method is based on semi-definite programming and requires rigorous centralized computation so it is not always suitable for many ad-hoc setups.

Some forms of ad-hoc localization also exist in the domain of mobile robotics [8,15]. The localization problem in mobile robotics bears many similarities to the ad-hoc localization problem investigated in sensor networks. One main difference however is that mobile robots have additional odometric measurements that can help with estimating the initial robot positions, something that is not available in sensor networks. Furthermore, localization studies in the sensor network community also consider scalability communication and power consumption issues that are not studied by the robotics community.

The AHLoS system we proposed in [17] uses iterative multilateration which relies on a small set of nodes initially configured as beacons to estimating node locations in an ad-hoc setup. This work identified two main problems: (1) iterative multilateration is sensitive to beacon densities and can easily get stuck in places where beacon densities are sparse, (2) error propagation becomes an issue in large networks. The collaborative multilateration algorithms presented in this paper address these two issues.

In parallel to our work some other ad-hoc node localization approaches have been independently proposed in [13] and [16]. In both these approaches anchor node location information is propagated across the network. When anchors become aware of other anchor node locations, they use this information to estimate the average hop length in their vicinity and broadcast it back into the network. Nodes with unknown locations also note the shortest hop distance to each of the anchor nodes and multiply it with the broadcasted average hop length to get an approximate distance to each of the anchor nodes. With this information nodes perform a multilateration to get an initial estimate of their locations. To obtain better estimates, the authors of [16] also use a further refinement phase that uses least squares to refine node positions based on local computation. Simulation studies have shown that these technologies are independent of ranging technologies and can deliver localization accuracy within one third of the communication range.

Despite the similarities, the work presented here has several fundamental differences from the aforementioned approaches. Error propagation is limited by computing in the context of over-constrained node configurations. We also provide a computation cost comparison between the centralized and distributed computation models and we examine the scalability of this approach by comparison to the Cramér–Rao bounds. Our algorithm development is tuned for a fine-grained ad-hoc localization system comprised of resource constrained sensor nodes equipped with an ultrasonic ranging system for measuring inter-node distances.

3. Preliminaries

3.1. Establishing local coordinate systems

The initial beacon locations can be derived from manual placement or by automatically establishing a local coordinate system. One method for establishing a coordinate system is to use a tiered system in which some nodes that are capable of accurate long distance ranging (i.e., long range ultrasound or laser range finders). These nodes establish a local coordinate system as shown in figure 1. The individual coordinate systems can then be merged into a global coordinate system. One possible method for this is described in [2]. In this paper we focus our attention in the fine-grained localization of nodes inside a local coordinate system (i.e., inside one of the triangles in figure 1).

3.2. Example sensor node and ranging considerations

Figure 2 shows our second generation node, the Medusa MK-2, a low cost wireless sensor node we have developed for experimenting with node localization problems. This node consists of a 4 MHz 8-bit AVRMega128L microcontroller from Atmel and a low power RFM radio, in a configuration that is similar to UC Berkeley's Mica motes. In addition, the MK-2 node carries a more powerful 40 MHz AT91FR4081 ARM THUMB coprocessor with 136 KB of RAM and 1 MB of on-board FLASH memory for more intensive computation tasks. The details and additional features of the MK-2 architecture are described in [19]. For localization, the node

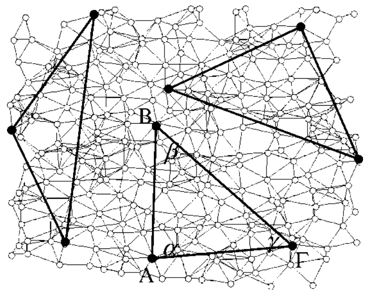


Figure 1. Establishing local coordinate systems.

Figure 2. The Medusa MK-2 experimental node.

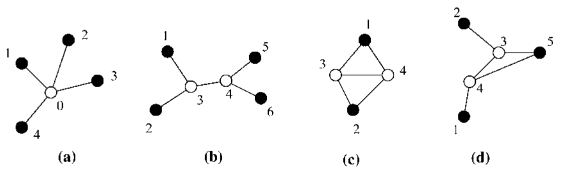


Figure 3. (a) One-hop multilateration, (b) two-hop multilateration, (c) symmetric case, (d) each unknown has one independent reference.

is equipped with 40 KHz ultrasonic sensors that have an effective range of 5 meters and approximately 1 cm accuracy. Similar technologies [6] can produce longer ranges, but we found this range to be more appropriate for indoor settings.

3.3. Solution outline

In a single hop setups, such as the one in figure 3, where nodes are within range from a set of beacons, nodes with unknown locations can determine their locations between measured and estimated distances to the beacons [17]. In this paper we consider a multilateration that spans over multiple hops. This enables nodes that are not directly connected to beacon nodes to collaborate with other intermediate nodes with unknown locations situated between themselves and the beacons to jointly estimate their locations. One of the main challenges in this problem is to prevent error accumulation inside the network. To prevent error accumulation, the node localization problem is set up a least squares estimation problem with respect to the global network topology. At the same time, the limited communication and computation resources and the need for robust behavior, suggest that a multilateration operation over multiple hops should also operate in a distributed manner.

Collaborative multilateration takes place in three main phases and a post-processing phase. In the first phase, the nodes self-organize into groups, *collaborative subtrees* so that nodes with unknown positions are over-constrained and can have only one possible solution. At this point, any nodes that do not satisfy the required constrains do not become part of any collaborative subtree. During the second phase, the nodes use simple geometric relationships between measured distances and known beacon locations to obtain a set of initial position estimates. The third phase is a refinement phase that uses iterative least squares to obtain the final position estimates. Finally, the post processing phase very similar to the second phase, and uses all the new location information to further refine the positions of under-constrained nodes.

4. Initial configuration

4.1. Phase 1: collaborative subtrees

A computation subtree constitutes a configuration of unknowns and beacons for which the solution to the position estimates of the unknowns can be uniquely determined. This is achieved by obtaining a well-determined or preferably over-determined set of equations – n variables to be estimated and at least n equations. Before attempting to solve these equations, the solution uniqueness is determined to prevent the estimation of erroneous locations. Collaborative subtrees have another desirable property that will become apparent when we discuss our distributed computation model in section 4.2. To determine the requirements for solution uniqueness we develop our discussion by reviewing the requirements of the single hop multilateration. Later on, we augment these requirements to cover the multihop case.

4.1.1. One-hop multilateration requirements

In the single hop setup of figure 3(a), the basic requirement for one unknown node to have a unique solution on a 2D plane is that it is within range of at least three non-collinear beacons. If the beacons lie in a straight line, the node configuration is symmetric, and there is more than one possible solution.

4.1.2. Two-hop multilateration requirements

Using the one-hop multilateration requirements as a starting point, the corresponding set of requirements for a two-hop multilateration can be established. A two-hop multilateration represents the case where the beacons are not always directly connected to the node but they are within a two-hop radius from the unknown node. In this situation, two or more unknown nodes can utilize the beacon location information and the intermediate distance measurements between themselves and the beacons to jointly estimate their locations. Like the one-hop case, each unknown node needs to be connected to at least three nodes, but these nodes are not required to be beacons. Instead, unknown nodes need to determine which of their neighbors have only one possible position solution and use them as reference points to determine if their position solution is unique. From this perspective, a position solution is tentatively unique if it has at least three neighbors that are either beacons or their solutions are tentatively unique. Figure 3(b) illustrates the most basic case. Nodes 3 and 4 are unknown and they are both connected to three nodes. Note that from the perspective of node 3, one of its links terminates to an unknown, node 4. Node 4, however, has two more outgoing links to beacons 5 and 6. If we assume that node 3 has a unique position solution, then node 4 also has a unique position solution. If, however, node 4 has a unique position solution, then node 3 is also collaborative because it is connected to 3 collaborative nodes – 1, 2 and 4. This condition is necessary but not sufficient to guarantee that there is only one possible node position estimate. Many symmetric topologies that meet the above requirement can yield more than one possible position estimate.

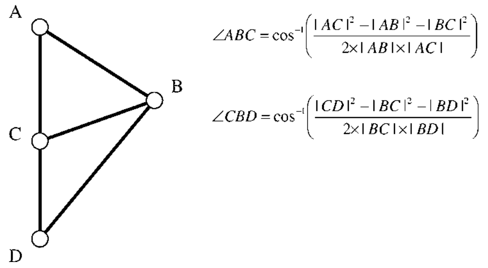


Figure 4. Detecting collinear configurations.

Condition 1. To have a unique possible position solution, it is necessary that an unknown node be connected to at least three nodes that have unique possible positions.

The fist symmetric case follows from the conditions of the single hop setup – the nodes with tentatively unique solutions used as references for an unknown should not lie in a straight line. If they lie in a straight line, then the unknown node will have two possible positions so the solution to the location estimate is not unique.

Condition 2. It is necessary for an unknown node to use at least one reference point that is not collinear with the rest of its reference points.

Although the positions of the reference points are not known, one can test for this condition using basic trigonometry. In figure 4, assuming that nodes A, C and D are known to have unique solutions, node B tries to establish if its position solution is unique. To do so node B computes the angles ABC, CBD and ABD. Using the angle ABD, node B can calculate the distance AD. If the computed distance AD is equal to the sum of distances AC and CD then the nodes are collinear[1] hence node B decides that its solution is not unique.

Another type of setup that can cause symmetry problems is shown in figure 3(c). Nodes 3 and 4 both have 3 links to nodes with tentatively unique positions but the setup is symmetric since the two nodes can be swapped without any violation of the constraints imposed by the intra-node distance measurements. To avoid this situation where the whole network can be rotated over two pivot points (nodes 1 and 2 in this example) an additional condition is set.

Condition 3. In each pair of unknown nodes that use the link to each other as a constraint, it is necessary that each node has at least one link that connects to a different node from the nodes used as references by the other node.

The network in figure 3(d) is an example configuration that satisfies this property. Both unknown nodes 3 and 4 have at least one independent reference. Node 4 has beacon 1 and node 3 has beacon 2. The above three conditions are individually necessary but jointly sufficient to guarantee that if an unknown node is within two hops from at least three beacons then the unknown has a single possible position solution.

[1] Here we loosely use the term 'equal' for clarity and simplicity of the explanation, in practice we also need to consider the noise incurred by the distance measurement process.

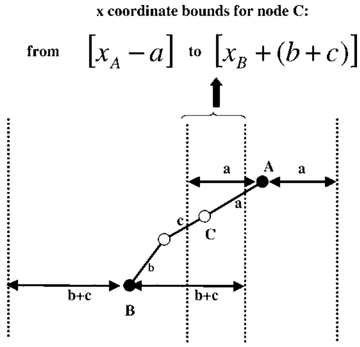


Figure 5. Initial estimates.

4.1.3. n-hop multilateration requirements

To determine if nodes located within n hops from the beacons have unique solutions a similar set of criteria is applied. Starting from an unknown node we test if it has at least three neighbors with tentatively unique positions. If the node has three neighbors that do not already know if their solution is unique, then a recursive call is executed at each neighbor to determine if its position is unique. To meet the requirements of condition 3 each node used as an independent reference is marked as used. This prevents other nodes from subsequent recursive calls to re-use that node as an independent reference. At every step, each node checks if the criteria for condition 2 are also met.

4.2. Phase 2: obtaining initial estimates

The initial estimates are obtained by applying the distance measurements as constraints on the x and y coordinates of the unknown nodes. Figure 5 shows how the distance measurement from two beacons A and B can be used to obtain the x coordinate bounds for the unknown node C. If the distance between an unknown and the beacon A is a then the x coordinates of node C are bounded by a to the left and to the right of the x coordinate of beacon A, $x_A - a$ and $x_A + a$. Similarly, beacon B which is two hops away from C, bounds the coordinates of C through the length of the minimum weight path to C, $b + c$, so the bounds for C's x-coordinates with respect to B are $x_B - (b + c)$ and $x_B + (b + c)$. By knowing this information C can determine that its x coordinate bounds with respect to beacons A and B are $x_B + (b+c)$ and $x_A - a$. This operation selects the tightest left-hand side bound from and the tightest right-hand side bound from each beacon. The same operation is applied on the y coordinates. The node then combines its bounds on the x and y coordinates, to obtain a bounding box of the region where the node lies. To obtain this bounding box, the locations of all the beacons are forwarded to all unknowns along a minimum weight path. This forwarding is the same idea as distance vector routing but using the measured distances instead of hops as weights.

The initial position estimate of a node is set at the center of the bounding box. To obtain a good set of initial estimates with this method we assume that some of the beacons must lie on the perimeter of the network.

5. Phase 3: position refinement

In the third phase, the initial node positions are refined, using least-squares estimation. Position refinement can be implanted in one of two possible computation models, centralized or distributed that are described next.

5.1. Computing at a central node

Using the collaborative subtrees and the initial position estimates, the unknown node position estimates can be computed at a central point. The edges of the computation subtree give a well-determined or over-determined set of equations, which can be solved using non-linear optimization. The non-linear of equations for the network in figure 3(b) is shown in equation (1).[2] As in the one hop case, the objective is to minimize the residuals between the measured distances between the nodes and the distances computed using the node location estimates.

$$\begin{aligned} f_{2,3} &= R_{2,3} - \sqrt{(x_2 - ex_3)^2 + (y_2 - ey_3)^2}, \\ f_{3,5} &= R_{3,5} - \sqrt{(ex_3 - x_5)^2 + (ey_3 - y_5)^2}, \\ f_{4,3} &= R_{4,3} - \sqrt{(ex_4 - ex_3)^2 + (ey_4 - ey_3)^2}, \\ f_{4,5} &= R_{4,5} - \sqrt{(ex_4 - x_5)^2 + (ey_4 + y_5)^2}, \\ f_{4,1} &= R_{4,1} - \sqrt{(ex_4 - x_1)^2 + (ey_4 - y_1)^2}. \end{aligned} \tag{1}$$

The $R_{i,j}$ quantities represent the measured distances between two nodes and the quantities under the square root indicate the estimated distances. $f_{i,j}$ represent the residual between the measured and estimated quantities. The objective function in (2) is to minimize the mean-square error over all equations; the difference of this from its one hop counterpart is that in this process, unknown–unknown links are also used as constraints:

$$F(x_3, y_3, x_4, y_4) = \min \sum f_{i,j}^2. \tag{2}$$

The solution to this optimization problem can be obtained using some of the standard least squares methods. Our implementation uses a Kalman filter [1,22], which provides the same location estimates as iterative least-squares in a static network [5]. The Kalman filter was chosen because of its ability to fuse measurements from multiple sensing modalities and to track the nodes after the localization process is complete.

5.2. Computing at every node

In the distributed version, of our algorithm, computation is spatially distributed across the network and each unknown node is responsible for computing its own location estimate. This is achieved by performing local computation and communication with the neighboring nodes. This idea is similar to using a distributed Kalman filter [14,15], but also has some differences. First, the Kalman filter executes in the context of a computation subtree and each node executes a one hop multilateration based on its distance measurements and the location information from its neighbors. Second, instead of using a decentralized Kalman filter, we use an approximation in which nodes do not exchange covariance information is used. This conserves energy since it reduces communication, and simplifies implementation. Third, the computation is driven by ad-hoc networking protocols.

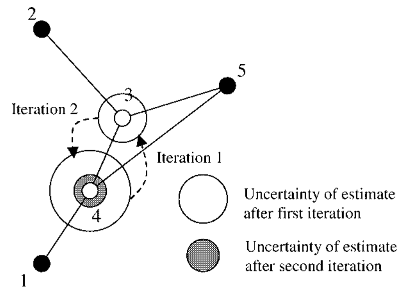


Figure 6. Distributed computation on a simple network.

The underlying principle of our distributed scheme is that after the completion of the first two phases, each node inside the computation tree computes an estimate of its location. Since most unknown nodes, are not directly connected to beacons, they use the initial estimates (obtained in section 3.2) of their neighbors as the reference points for estimating their locations. As soon as an unknown computes a new estimate, it broadcasts this estimate to its neighbors, and the neighbors use it to update their own position estimates. This computation is repeated from node to node across the network until all the nodes reach the pre-specified tolerance Δ, represented by the convergence gradient. Figure 6 is a pictorial representation of the computation process.[3] First node 4 computes its location estimate using beacons 1 and 5 and node 3 as references. Once node 4 broadcasts its update, node 3 recomputes its own estimate using beacons 2 and 5 and the new estimate received from node 4. Node 3 then broadcasts the new estimate and node 4 uses this to compute a new estimate that is more accurate than its previous estimate.

If this process proceeds uncontrolled, then the nodes will converge at local minima and erroneous estimates will be produced. Imagine a computation subtree with many unknown nodes (i.e., 20). If two neighboring unknown nodes A and B that compute and broadcast their updates as soon as an update from each other is received, then their updating process will proceed faster than the remaining nodes in the computation subtree. This introduces a “local oscillation” in the computation that makes the nodes converge to their final estimates much faster but without complying with the global gradient, thus yielding erroneous estimates.

[2] The prefix e in front of x, y denotes estimated coordinates.

[3] Note that in practice the position uncertainties are represented by ellipses rather than circles.

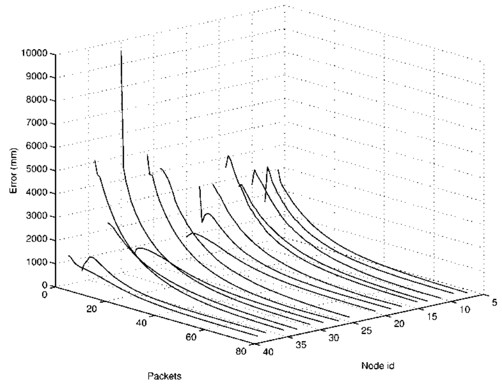


Figure 7. Progress of distributed computation.

To prevent this problem, the multilaterations at each node are executed in a consistent sequence across all the unknown members of the computation subtree. This sequence is repeated until the multilaterations of all the members of the computation subtree converge to the pre-specified tolerance. The in-sequence execution of the multilaterations inside the computation subtree establishes a gradient with respect to the global topology constraints at each node, thus enabling the node to compute its global optimum locally.

Figure 7 is an excerpt from our combined ns-2–MATLAB simulation which demonstrates the execution of distributed position refinement on a network of 34 nodes and 6 beacons.[4] The unknown nodes in the network have an average degree of 4, 15 meters[5] range and the 20 mm white-Gaussian noise in the distance measurement system. The x-axis shows the number of packets (estimate updates) transmitted by each node, the node ids are shown in the y-axis and the z-axis shows the error in millimeters. The values of the error before any packets are transmitted, at packets $= 0$ on the x-axis, represent the errors of the initial estimates (obtained in section 3.2). As it can be seen form the figure, each node starts at different levels of error. After a few iterations of executing the node sequences in the computation subtree, a global gradient is established that drives the error down across the whole subtree. In the end, each node succeeds in estimating its node location with a 3-centimeter accuracy. Error accumulation is prevented by the global constraints of the collaborative subtree.

The order of nodes executing in the computation subtree sequence does not need to be specified but it needs to be consistent over successive iterations of the sequence. This entails that the order with which nodes compute their position updates has to be consistent across iterations. One possible way to initiate this distributed computation process is to use Distributed Depth First Search (DDFS) [21]. DDFS search is started at an arbitrary unknown node within the computation tree and it runs for two iterations. During the traversal of the subtree by DDFS, when each node is marked visited, the node it computes and broadcasts its location estimate and starts a timer. In the second iteration of DDFS, nodes compute and broadcast their locations. At this point the nodes also stop the previously set timer. The time between the two visits denotes the time interval at which each node should recompute and broadcast a new position update.

[4] For clarity and good visibility purposes the graph only shows how the process proceeds on the even numbered nodes.

[5] In our actual node testbed this is reduced to 5 meters to facilitate multiple hops in a lab setting.

6. Evaluation

We evaluate the performance of the collaborative multilateration through a set of simulations. The Kalman filters are implemented in MATALB and they are linked into the ns-2 simulator using the MATLAB compiler and the MATLAB C++ library. The required protocols for communication are implemented inside ns-2. Using this simulation setup we carried out a series of experiments on a test suite of 200 different scenarios. Our simulation parameters are set to match the parameters of our experimental node. Each node has an effective radio range of 15 meters and a raw data rate of 20 kbps. Each node can measure distances between its neighbors with the same range as the radio. For the centralized implementation we also used DSR as the routing later. The measurement noise is modeled as a zero-mean Gaussian random variable with 20 mm standard deviation.

6.1. Computation cost comparison

Our first experiment compares the computation overhead between the distributed and centralized computation methods by recording the number of FLOPS consumed by MATLAB to compute the position estimates in each case. The scenarios used for this test have 6 beacons and varying number of unknowns ranging from 10 to 100 nodes. The number of unknowns was used in increments of 10, and the results show the average for 20 scenarios of each type. In all cases the network density is kept constant and each node has an average of 6 neighbors. The cumulative number of MFLOPS for the centralized and distributed implementation are shown in figure 8. From this result, we found that the computation overhead of the centralized computation model increases fast with the number of unknown nodes. In this particular test, the computation overhead appears to be cubic with the number of nodes. The distributed computation model on the other hand scales linearly with the number of nodes. The slope for

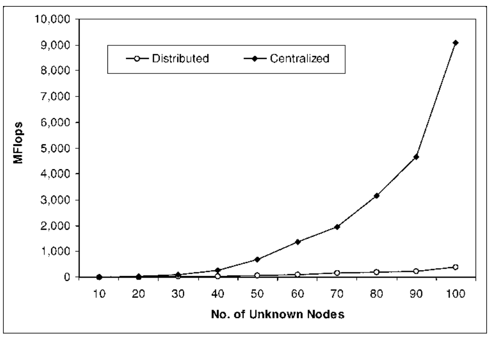


Figure 8. Computation cost comparison.

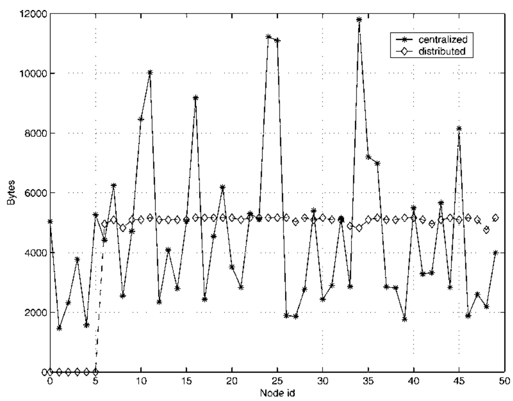


Figure 9. Communication cost on a 50 node network (6 beacons, 44 unknowns).

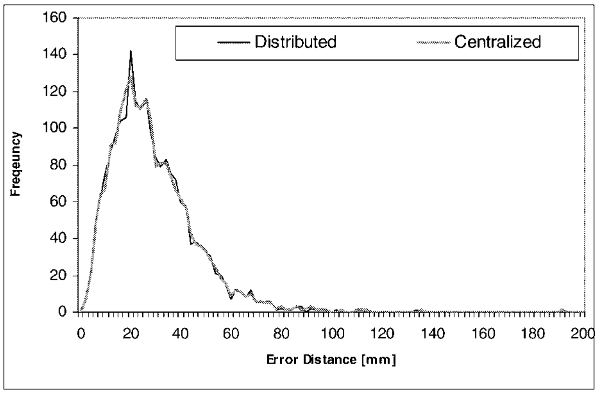


Figure 10. Error distribution for estimated locations.

the distributed case in figure 8 is 3.7 Mflops, meaning that each node spends approximately 3.7 Mflops to compute an estimate of its location. This makes localization feasible on small resource constrained nodes such as the MK-2 node described in section 3.2 feasible.

6.2. Communication cost and convergence latency considerations

The convergence latency and communication aspects of collaborative multilateration are more difficult to evaluate because of their dependence on multiple system attributes. In the fully distributed case, convergence latency depends on the available communication bandwidth and the node processing power. Convergence latency also depends on the size of the computation tree. As the number of nodes increases, the sequence of Kalman filter executions will take longer to complete and more iterations of the sequence are required. The communication pattern is uniform across all the nodes.

Evaluating the communication cost is more complex. In a clustered architecture, the communication cost depends on the cost of electing a cluster head and the routing cost for propagating the information back and forth from the cluster head. A notable trend in communication cost is shown in figure 9. The average number of bytes transmitted is 4596 for the centralized scheme and 4485 for the distributed scheme. Although on average the communication cost is almost the same, the distributed scheme has an even distribution of transmitted bytes.

6.3. Localization accuracy

Our simulations evaluate the accuracy of the localization process based on the measurement noise parameters of our ultrasonic distance measurement system. Figure 10 shows the cumulative error distribution over all scenarios used in this experiment for both the distributed and centralized cases. In both cases the average error was 27.7 mm with a standard deviation of 16 mm. These results are also consistent with simple topologies tested on our localization testbed.

6.4. Comparison to the Cramér–Rao bounds

To further evaluate the scalability and performance of collaborative multilateration, we performed a comparison with the Cramér–Rao (CR) bound. The CR bound is a classical result from statistics that gives a lower bound on the error covariance matrix for an unbiased estimate of a certain parameter. For our purposes we have developed the CR bound for a multihop localization setup by assuming a zero-mean Gaussian error distribution [18]. Using this result, we run a set of simulations to test the scalability of multihop localization and to compare our collaborative multilateration algorithms against the theoretical bounds.

Figure 11 indicates how the RMS localization error bound behaves as the number of nodes with unknown locations increases, at a constant density of 6 neighbors per node and a constant percentage of beacons set to 10%. The resulting bounds indicate that error propagation can be contained as the number unknowns in the network scales. By repeating the same simulation over different measurement error variances we note that similar behavior can be expected for different ranging technologies other than ultrasound.

Finally, in figure 12 shows a comparison of the collaborative multilateration algorithm to the CR bound. From this comparison, both the centralized and distributed computation models of collaborative multilateration closely follow the bound but we also notice some differences. First, the critical density breakpoint is shifted from 6 neighbors per node to 8 neighbors per node. Second, the distributed computation model appears to diverge at some points. A closer examination has shown that this is attributed to isolated scenarios. This divergence can be eliminated by enforcing a series of consistency checks that restart the distributed multilateration at a different point in the network when an inconsistent gradient is detected. At higher densities (12 neighbors or more), the two computation models give consistent results. This is attributed to the increased network redundancy that prevents isolated cases from diverging. It also implies that at large density deployments, consistency checking can be suppressed in the interest of implementation complexity and power conservation.

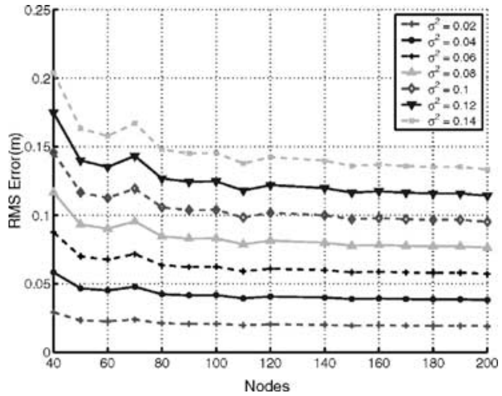


Figure 11. Error propagation as network scales at 10% beacons, 6 neighbors per node.

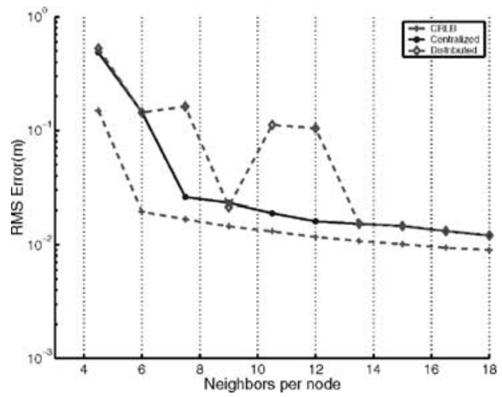


Figure 12. Collaborative multilateration comparison to the Cramér–Rao bound.

7. Conclusions and future work

In this paper we have described collaborative multilateration for node localization problems. We have shown that using this three phase approach nodes that are indirectly connected to beacon nodes can estimate their locations with similar accuracies at the single hop multilateration. Also, with our distributed approach colonies of resource constrained sensor nodes can collectively solve a global optimization problem that an individual node cannot solve. The use of a global gradient for computing a global optimum locally reinforces a distributed computation model with other potential applications in sensor networks. In addition to the distributed computation model, the collaborative multilateration appears to be an attractive choice for assisting infrastructure based localization systems to better handle obstructions. A comparison to the bounds has shown that localization of nodes over multiple hops would be scalable to larger networks. The work in this paper focuses on the computational aspects of collaborative multilateration. The remaining challenge is to study its feasibility with respect to the physical effects. To this end, as part of our future work we plan to study the interaction of our algorithms with the physical world using our sensor network testbed of Medusa MK-2 nodes.

Acknowledgements

This paper is based, in part, on research funded through NSF under Grant number ANI-008577, and through DARPA SensIT and Rome Laboratory, Air Force Material Command, USAF, under Agreement number F30602-99-1-0529. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the NSF, DARPA, or Rome Laboratory, USAF.

References

[1] R. Brown and P. Hwang, *Introduction to Signals and Applied Kalman Filtering* (Wiley, New York, 1997).

[2] S. Capkun, M. Hamdi and J.P. Hubaux, GPS-free positioning in mobile ad-hoc networks, in: *Hawaii International Conference on System Sciences, HICCSS-34* (January 2001).

[3] L. Doherty, L. El Ghaoui and K.S.J. Pister, Convex position estimation in wireless sensor networks, in: *Proceedings of InfoCom 2001*, Anchorage, AK (April 2001).

[4] D. Estrin, R. Govindan, J. Heidemann and S. Kumar, Next century challenges: scalable coordination in sensor networks, in: *Proceedings of the Fifth Annual International Conference on Mobile Computing and Networking*, Seattle, Washington (1999) pp. 263–270.

[5] E. Foxlin, M. Harrington and G. Pfeiffer, Constellation(tm): a wide-range wireless motion-tracking system for augmented reality and virtual set applications, in: *Proceedings of SIGGRAPH 98*, Orlando, FL, July 19–24 (1998).

[6] L. Girod and D. Estrin, Robust range estimation using acoustic and multimodal sensing, in: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2001)*, Maui, HI (October 2001).

[7] J. Hightower and G. Boriello, Location systems for ubiquitous computing, *IEEE Computer* 34(8) (2001) 57–66.

[8] A. Howard, M.J. Mataric and G.S. Sukhatme, Relaxation on a mesh: a formalism for generalized localization, in: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS01)* (2001) pp. 1055–1060.

[9] Intersense Inc., `http://www.isense.com`

[10] E. Kaplan, *Understanding GPS Principles and Applications* (Artech House, 1996).

[11] M. Mauve, J. Widmer and H. Hartenstein, A survey on position based routing in mobile ad-hoc networks, IEEE Network Magazine 15(6) (2001) 30–39.

[12] S. Meguerdichian, F. Koushanfar, G. Qu and M. Potkonjak, Exposure in wireless ad hoc sensor networks, in: *International Conference on Mobile Computing and Networking (MobiCom'01)*, Rome, Italy (July 2001) pp. 139–150.

[13] D. Nicolescu and B. Nath, Ad-hoc positioning system, in: *Proceedings of IEEE GlobeCom* (November 2001).

[14] B.S. Rao and H.F. Durrant-Whyte, Fully decentralized algorithm for multisensor Kalman filtering, IEE Proceedings D 138(5) (September 1991).

[15] S.I. Roumeliotis and G.A. Bekey, Synergetic localization for groups of mobile robots, in: *Proceedings of the 39th IEEE Conference on Decision and Control*, Sydney, NSW, Australia, 12–15 December 2000, Vols. 4, 5 (IEEE, Piscataway, NJ, 2000) pp. 3477–3482.

[16] C. Savarese, J. Rabay and K. Langendoen, Robust positioning algorithms for distributed ad-hoc wireless sensor networks, in: *USENIX Technical Annual Conference*, Monterey, CA (June 2002).

[17] A. Savvides, C.C. Han and M.B. Srivastava, Dynamic fine-grained localization in ad-hoc networks of sensors, in: *Proceedings of the Seventh*

Annual International Conference on Mobile Computing and Networking (MobiCom), Rome, Italy (July 2001) pp. 166–179.

[18] A. Savvides, A. Sachin, W. Garber, R. Moses and M.B. Srivastava, On the error characteristics of multihop node localization in ad-hoc sensor networks, in: *Proceedings of the Second International Workshop on Information Processing in Sensor Networks*, April 22–23 (2003).

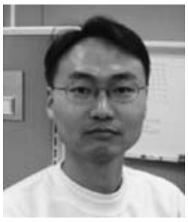
[19] A. Savvides and M.B. Srivastava, A distributed computation platform for wireless embedded sensing, in: *Proceedings of ICCD 2002*, Freiburg, Germany, to appear.

[20] M.B. Srivastava, R. Muntz and M. Potkonjak, Smart Kindergarten: sensor-based wireless networks for smart developmental problem-solving environments, in: *Proceedings of the Seventh Annual International Conference on Mobile Computing and Networking* (MobiCom 2001), Rome, Italy (July 2001) pp. 132–139.

[21] G. Tel, Distributed graph exploration, available at `http://carol.wins.uva.nl/delaat/netwerken_college/explo.pdf`

[22] G. Welch and G. Bishop, An introduction to the Kalman filter, available at `http://www.cs.unc.edu/welch/kalman/kalmanIntro.html`

Andreas Savvides received his B.S. degree in computer engineering from the University of California, San Diego (UCSD) in 1997 and his M.S. in computer engineering from the University of Massachusetts, Amherst in 1999. He is currently a Ph.D. candidate in electrical engineering at the University of California, Los Angeles (UCLA). Andreas is also a member of the Networked and Embedded Systems Lab (NESL) and the NSF Science and Technology Center on Embedded Networked Sensing (CENS). Prior to joining UCLA, Andreas worked ad-hoc networking at the HRL Labs in Malibu, California. His research interests are in sensor networks, embedded systems and ubiquitous computing. He is also a student member of IEEE.
E-mail: asavvide@ee.ucla.edu



Heemin Park received B.S. and M.S. degrees in computer science from Sogang University, Seoul, Korea in 1993 and 1995, respectively. Since then, he has worked for Samsung Electronics, Co., Ltd., Korea. Since September 2001, he has been pursuing a Ph.D. degree in electrical engineering at University of California, Los Angeles. His research interests include design of networked and embedded computing systems, and wireless sensor networks.
E-mail: hmpark@ee.ucla.edu

Mani B. Srivastava received the B.Tech. in electrical engineering from IIT Kanpur in India, and the MS and Ph.D. degrees from the University of California at Berkeley. He is an associate professor of electrical engineering at the University of California, Los Angeles. From 1992 through 1996, he was a member of the technical staff at Bell Laboratories in Networked Computing Research. His current research interests are in mobile and wireless networked computing systems, low power systems, and sensor networks. He received the US National Science Foundation CAREER Award in 1997, and the President of India Gold Medal in 1985. He is a senior member of IEEE.
E-mail: mbs@ee.ucla.edu