EE290N Class Project

Convex Position Estimation in Distributed Sensor Networks

Lance Doherty
Berkeley Sensor and Actuator Center
Advisor: K.S.J. Pister
December 15, 1999

Smart Dust Overview

The Smart Dust project aims to scale sensing communication platforms down to cubic millimeter volumes. Combining CMOS technology for processing logic, MEMS corner-cubes for passive communication and thick-film batteries for power, Smart Dust exploits current technological mass production methods. With the ability to economically produce thousands of sensor nodes (known as Smart Dust “motes”), local information can be gathered from a wide geographical area. As a showcase for the capabilities of the goal product, a macroscopic breed of Smart Dust has been developed on printed circuit boards with commercial off-the-shelf products for sensing, transmission and power. With active communication capabilities (RF, visible laser or IRDA), these “macromotes” allow for a more diverse set of network functions.

Different breeds of macromotes are currently in use. The RF-communicating mote has a transmission range of 20m in open space (figure 1) at 4800bps. Visible laser motes have been tested at distances of 20km. A scanning apparatus allows the beam to scan over a fraction of the hemisphere and two motes to locate each other. All current macromotes use Atmel microprocessors that are programmed via a PC serial port.

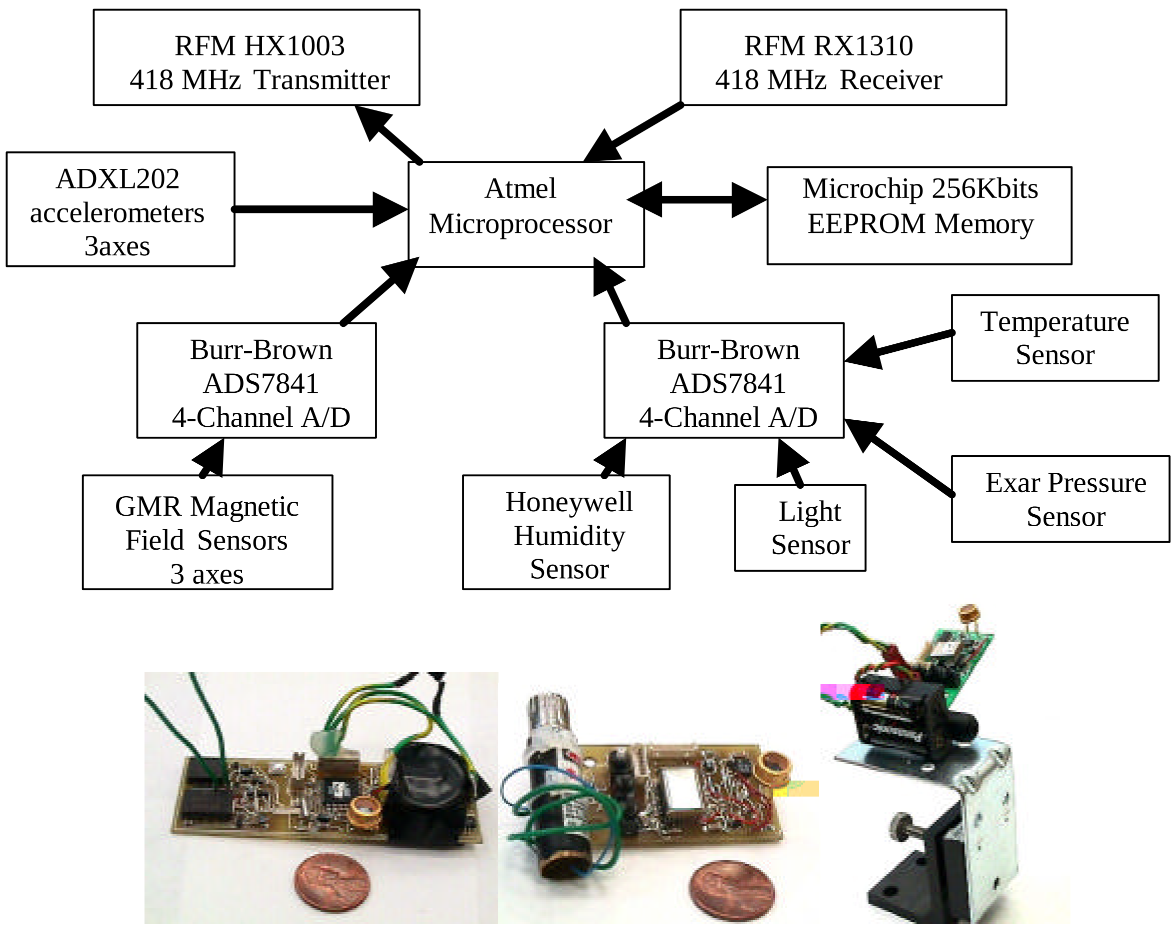

Figure 1: From left to right, RF, laser and mounted laser motes. Above the photos is a block diagram of the components of the RF mote. Courtesy of Seth Hollar.

A PC interface to the Smart Dust network provides a means of observing network behavior. In the current theme, the PC acts as an observer receiving data on which motes have a communication link between them. In graph theoretic terms, the PC learns which nodes are connected in the network graph. This information is then used to solve the position estimation problem.

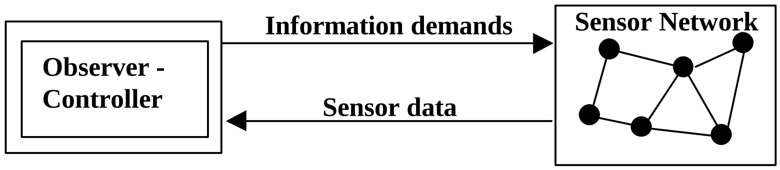

Figure 2: Flow of information. All network messages originate and terminate at the observer-controller. Each sensor mote has a unique ID and only local topological knowledge.

Position Estimation

In a network of thousands of motes, it is unlikely that the position of each mote will be determined by the designer. In the extreme case, motes may be dropped from the air and scatter about an unknown area. To process sensor data, however, it is important to know where the data is coming from. Motes could be equipped with a global positioning sensor (GPS) to provide them with knowledge of their absolute position, but this is currently a costly (in both volume and money) solution. Instead, positional information can be inferred from connection-imposed proximity constraints. In this model, only a few nodes have known positions (perhaps equipped with GPS or placed deliberately) and the remainder of the mote positions are computed.

This paper attempts to find feasible solutions to the position estimation problem using convex optimization techniques. In the network below, the red nodes represent known positions (data) while the blue node positions are estimated (variables). If one node can communicate with another, there exists a proximity constraint between them. For example, if a particular RF system can transmit 20m and two nodes are in communication, their separation must be less than 20m. These constraints restrict the feasible set of blue node positions. Only planar networks are considered, but augmenting them to 3-D is trivial. For emphasis, the position estimation problem is stated again:

Given: positions of red nodes
Find: a possible position for each blue node
Subject to: proximity constraints imposed by known connections

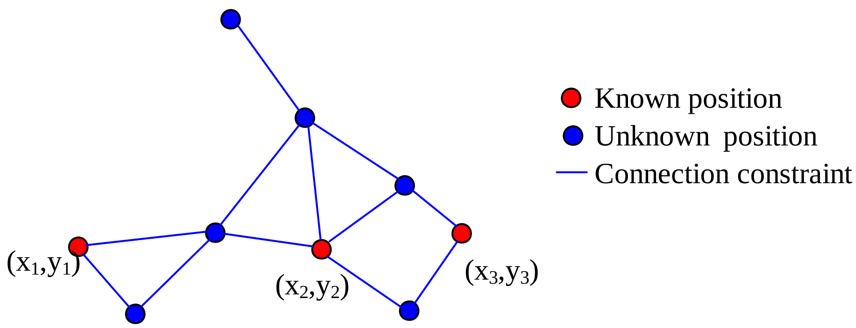

Figure 3: Graph illustrating data and variables as nodes, constraints as spans.

Note that there are constraints among blue nodes although their positions are unknown.

Constraint Models

This section outlines physical communication systems used in Smart Dust and explores the corresponding sets of constraints that can result from connections in the network. In the illustrations, the red nodes represent known positions and the blue node positions are unknown. The yellow area represents the allowable positions of the blue node.

As introduced above, an RF communication system can be modeled as having a rotationally symmetric range. In this model, a connection between nodes can be represented by a 2-norm constraint on the node positions. More specifically, for a maximum range R and node positions a and b, the equivalent LMI is given by:

$$\|a-b\|_2 < R \Rightarrow \begin{bmatrix} I_2 R & a-b \\ (a-b)^T & R \end{bmatrix} \geq 0 \quad (xR)$$

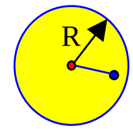

Figure 4: Mathematical and geometrical interpretation of the "xR" constraint.

This is a 3x3 matrix constraint that can be solved using semidefinite programming (SDP). A collection of all the connections in the network results in a larger SDP that can be solved using the same methodology. A more precise model uses the actual radius of each connection as an upper bound in the LMI:

$$r_{ab} \equiv \|a-b\|_2; \quad \begin{bmatrix} I_2 r_{ab} & a-b \\ (a-b)^T & r_{ab} \end{bmatrix} \geq 0 \quad (xr)$$

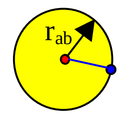

Figure 5: The "xr" constraint places the blue node on the perimeter of the feasible set.

Physically, this could be obtained by transmitters varying their output power during an initialization phase. If a connection is first obtained at a power P_o, this can be used to determine a tighter upper bound on each individual connection in the network. Note that neither of the following are convex constraints:

$$\|a-b\|_2 = r_{ab}; \quad \|a-b\|_2 > R$$

The latter would be useful in "pushing away" nodes that are not connected in the algorithm. This constraint is not physically realistic either – nodes within a certain range may not be able to communicate due to a physical barrier or transmission anisotropy. Formulating the former into a set of convex constraints with robustness would be a useful exercise.

In either infrared or laser reception, sensors are directional. With four sensors facing in the cardinal directions, a receiver knows not only that it is connected to another node (proximity constraint as above), but also in which quadrant the other node resides. For example, if node b is in the 4th quadrant relative to node a, the following constraints are obtained:

$$\begin{bmatrix} I_2 R & a-b \\ (a-b)^T & R \end{bmatrix} \geq 0, \quad \begin{cases} a_x - b_x \leq 0 \\ a_y - b_y \geq 0 \end{cases} \quad (xqR)$$

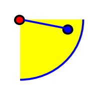

Figure 6: One LMI and two scalar inequalities are required to specify an "xqR" constraint.

The same technique of using the individual r_{ab}'s with quadrant information is also modeled and called the xqr system.

A more precise (and less realistic) directional model assumes exact knowledge of the angle between connected nodes. In this case, the blue node can be at any distance less than R along a specified angle from the red node. This is equivalent to the following constraints:

$$\left. \begin{cases} b_x = a_x + r\cos q \\ b_y = a_y + r\sin q \end{cases} \right\}, \quad r \in [0, R] \quad (xtR)$$

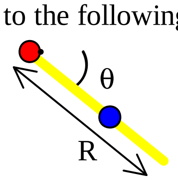

Figure 7: The blue node must lie on a certain line in the "xtR" constraint.

Of course, the individual radii can be used as in the previous examples resulting in the xtr system. This LP yields the exact positions of all nodes except in pathological cases and is of little research interest since it assumes too much accuracy in sensor information.

Adding uncertainty to the previous directional model gives another interesting possibility. Instead of θ being exactly known, suppose it is known to within some interval $[\theta\text{-}d\theta\ \theta\text{+}d\theta]$. Similarly, the distance r_{ab} is known to within some interval $[r_{ab}\text{-}dr\ r_{ab}\text{+}dr]$. This added uncertainty is illustrated below along with its convex (intersection of half-spaces) outer approximation.

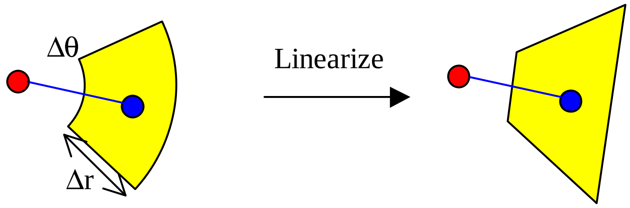

Figure 8: The "xc" constraint approximates a non-convex section of a donut.

This model is most relevant to an active beam steering system where the receiver can be rotated freely. There are four scalar linear constraints for each connection in this model, the form of which is dependent on the quadrant of θ and is too tedious to write out in detail in this report. This system is an LP and called xc (for cone constraint).

Table 1: Summary of different constraint methods.

System	Proximity	Directional	Physical	LMI constraints	Scalar constraints
xR	Constant	None	RF	1	0
xr	Varies	None	RF	1	0
xqR	Constant	Quadrant	IR/laser	1	2
xqr	Varies	Quadrant	IR/laser	1	2
xtR	Constant	Exact Angle	Laser	1	2
xtr	Varies	Exact Angle	Fiction!	0	2
xc	Constant	Angle Interval	Steering Laser	0	4

The effect of having more than one connection to a node is one of reducing the measure of the feasible set. The intersection of convex sets will itself be a smaller convex set.

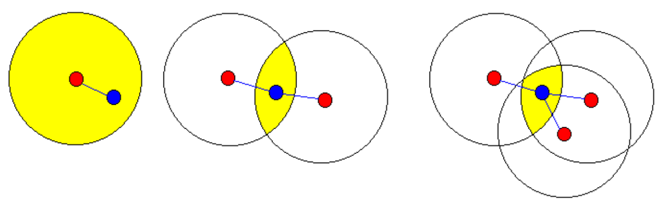

Figure 9: From left to right – the blue node has 1, 2 and 3 connections. Each additional connection effectively reduces the permissible position of the blue node.

Testing Procedure

The physical system requires that there exist a communications pathway from each node in the network to the external observer. To this end, only fully connected networks are considered. Random networks with n nodes are generated using the following algorithm[1]:

[1] All simulation is performed in the MATLAB environment

1. Place a node at (0,0)
2. Randomly select a placed node, a θ in [0 2π] and r in [0 R]
3. Place a node at this r and θ from the node in step 2
4. Repeat steps 2 and 3 (n-2) more times

This ensures that the network is fully connected, but results in an artificially dense network – the average connectivity is higher than would be expected if nodes were dumped at random (x,y) coordinates. Further testing could be done by extracting connected subnetworks from randomly placed nodes, but this would not alter the results presented in the sequel. R is always chosen to be identically 1 in arbitrary units.

Having generated a network, the following steps are carried out:

1. Find all connections (||a-b|| < R) and store distances in a matrix
2. Compute angle and/or quadrant information
3. Select a number of fixed (GPS-equipped) nodes, gnum, from the placed nodes
4. Choose a constraint system from the table above
5. Solve the position estimation problem for all n-gnum unknown positions simultaneously

Steps 1 and 2 are not required in a real network – this data would all be communicated directly to the observer. Step 5 uses whichever of LP or SDP[2] is appropriate.

Results

A 15-node network is generated[3] and kept for the duration of the experiment.

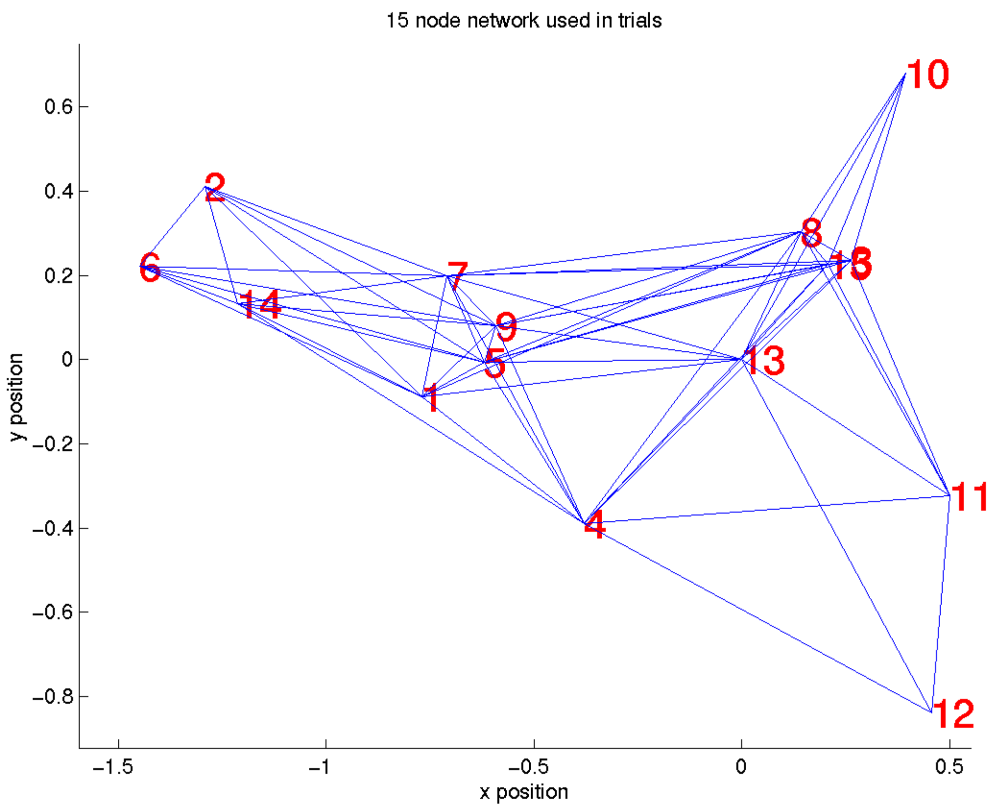

Figure 10: Positions of 15 nodes with all connections of length less than 1 illustrated.

[2] Generated with the aid of LMItool

[3] This required about 100 LMI constraints – the SDP solver crashed when presented with larger problems

For each of the first six (xR à xtr) constraint systems above, the number of fixed nodes (gnum) is increased from 2 to 14 and the position problem solved at each step for the n-gnum uncertain nodes. For each trial, the computed positions represent a member of the feasible set of node positions. The Euclidean proximity of this member to the actual positions is taken as a metric for the validity of the results. A larger discrepancy between the calculated and real positions means that the method is less effective.

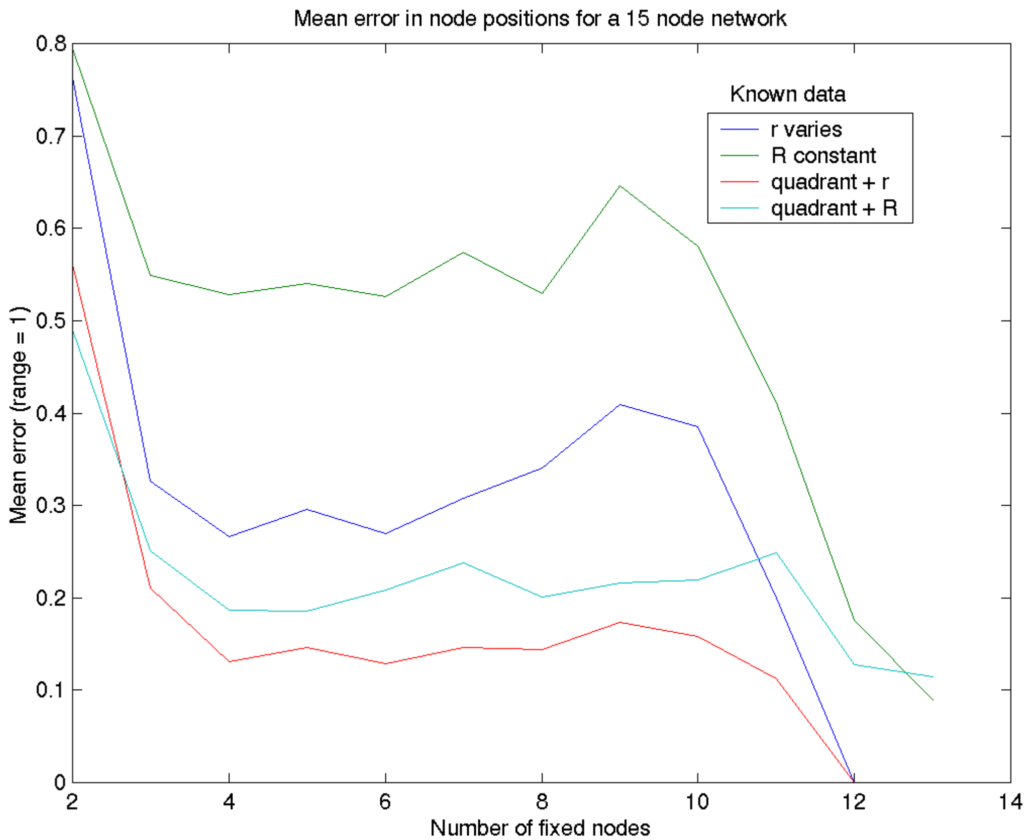

Figure 11: Performance of the SDP solution to a 15-node position estimation problem.

Data for xtR and xtr is not shown on the plot since both solve the problem perfectly for all values of gnum. As expected, methods with tighter constraints give feasible solutions that are generally closer to the real positions. Interestingly, gnum does not strongly affect performance between 3 and 10 indicating that constraints between blue nodes are as effective as blue-red constraints.

The added linear constraints for the quadrant method (over the simple xR and xr methods) require slightly more computation.

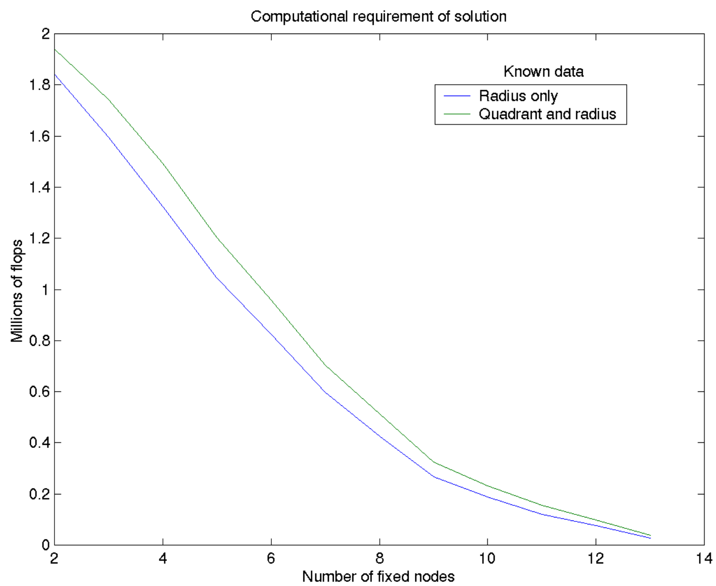

Figure 12: Required floating-point operations to solve the position estimation problem for 15 nodes.

As a test of the effects of connectivity on the discrepancy between computed and real positions, the xr method is run on 100 random 15-node networks with gnum = 4. This yields 11 computed positions for each network (or 1100 total). The experiment counts the number of connections to each node and plots this versus the error in the placement of that node.

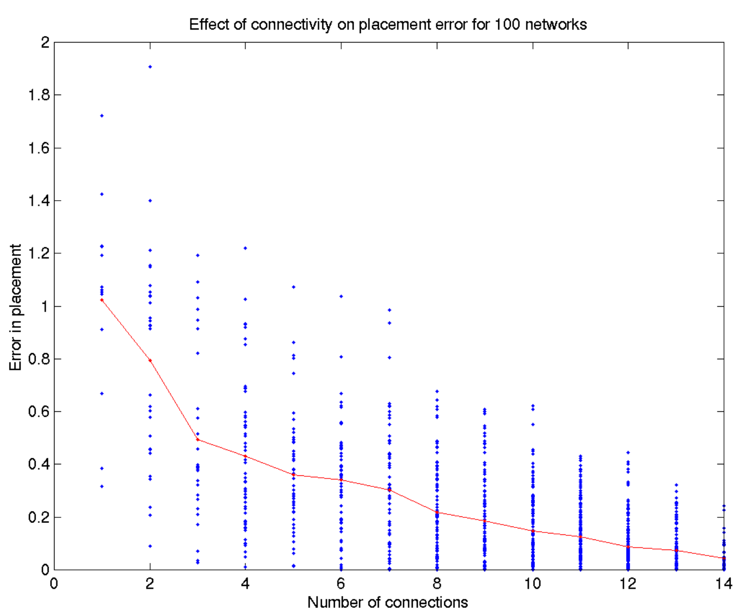

Figure 13: Each dot represents one node. The red line connects average error for each level of connectivity.

With one connection, the feasible set for the node consists of the entire disc around the connected node. It is reasonable that the mean error in this case would be the radius of the disc (1 in this case). The monotonic descent of the line indicates that more information (i.e. more connections) results in smaller feasible sets and hence more accurate estimation of position. A peripheral observation is the large connectivity of the network – few nodes are isolated with only one connection while many have more than 10 connections (recall that this is an artifact of the network generation algorithm).

A more thorough exploration of sensor uncertainty is possible using the xc method. This permits variation in dr and dθ to determine how precise the sensors must be to reconstruct the network.

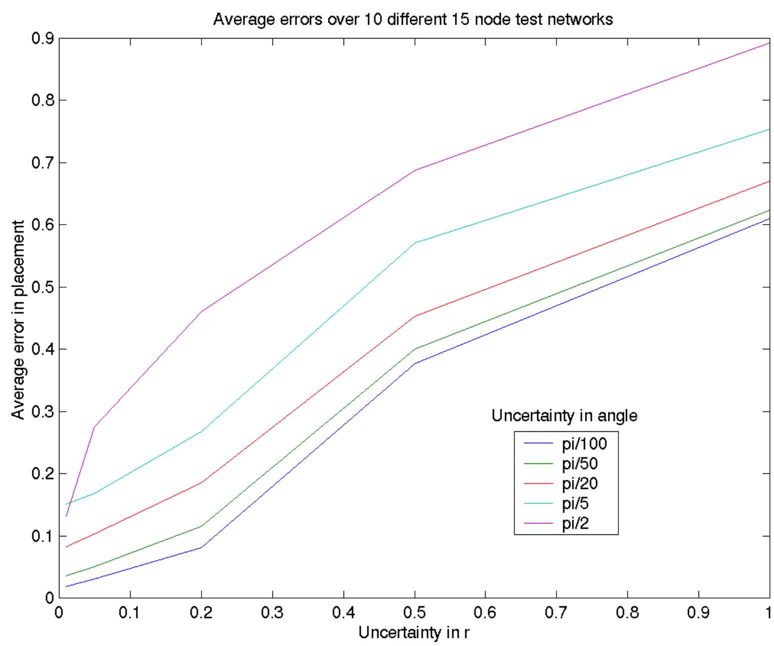

Figure 14: Data from 10 networks of 15 nodes using the xc method (gnum=4). The uncertainty in r is relative to the communication range set to 1. The uncertainty in q is given in radians.

As anticipated, placement error increases with both the uncertainty in r and θ. The change in performance due to angular uncertainty is far less pronounced than that suffered due to distance uncertainty. Doubling the angular uncertainty from π/100 to π/50 results in only a slight degradation while the error appears almost linear in the distance uncertainty.

Conclusions

Convex optimization techniques, namely LP and SDP, have been employed to reconstruct geographical position of nodes in a sensor network with a known set of proximity and/or directional constraints. The main result is that the methods presented all find a feasible solution to the position estimation problem. The relatively small errors incurred between calculated and real positions indicate that these methods are viable for position estimation in the dense sensor setting (i.e. with high connectivity). Using current technology, an RF system with R=20m could be used in the algorithms above. With microfabricated motes, we expect to have beam steering capable of discerning angle to within 1 mrad of angular deviation and laser systems that can approximate distance to within 10%. Using the xc method, this is sufficient to reconstruct the global positions to within about 1%. These techniques are directly applicable to planned Smart Dust communication schemes.

With the heavy computational load at 15 nodes, it is not clear that the system can be scaled to solve more complex position estimation problems. In particular, the number of flops required for solution using xc increases exponentially with the total number of connections. In the xr case, the computation time increases linearly with the number of connections if the total number of nodes is kept constant – it also increases with the total number of nodes. One solution is to ignore any more than m connections per node. Above about 5 connections, the nodes were placed accurately – to consider more connections may lead to unnecessary (and nearly redundant) constraints for the level of accuracy required.

In light of the data, it is postulated that the number of fixed nodes is not a strong factor in overall performance. This surprising result indicates that connection information among nodes with unknown positions is equally valuable as direct connections with fixed nodes. Providing that fixed nodes are situated near the perimeter of the network, the constraints serve to "stretch out" the connections within the network boundaries. However, if all the fixed nodes are placed at the center of the network, the estimated positions of the other nodes will collapse to the center as well (except in the xc model which has a mechanism for "pushing" nodes away). This suggests a placement scheme for a physical network and for future work.

Having executed the algorithm to estimate positions, these methods are useful also in the domain of sensing. Consider the scenario of tracking a body through a sensor network. Assume that the positions of all nodes have been estimated as above and are now fixed. Now let the moving body be a blue node at each time step; solve the position estimation problem for this single node. It has been shown that this can be done to high accuracy with enough connections – this translates to having a sufficient density of sensors in the network. We have also thus generated an algorithm for tracking in the dense sensor setting.

In the case that the network is dynamic and that all required information can be communicated to the external observer, perhaps the problem can be augmented to include a time dimension. In the solution of this problem (using similar methods to the position estimation problem), trajectories, not positions, would be the output.

The algorithm currently solves the network positions simultaneously. The problem can be broken up into smaller position estimation problems that are solved more rapidly. Once sections of the network are estimated, a hierarchical placement of these subnetworks can decide on the final positions. This methodology would reduce the accuracy of node placement – the parallel decision making process used in this project finds a globally feasible solution that may not be obtainable by separating the network into smaller problems.

Finally, there has been no attempt to implement an objective function to select the most likely blue node positions from the feasible set. One method for the xR constraint might be minimize the difference between the average of computed connection distances and the expected average distance based on the distribution. In a random distribution, the average connection length should be 2R/3 – hence we could minimize:

$$\left\| \frac{2R|C|}{3} - \sum_{C} r_{ij} \right\|$$

Where |C| is the number of connections. Another alternative, to obtain a measure of the feasible set, would be to bound each node's feasible set with an ellipsoid. The dimension of this ellipsoid could be used as positional variance in data recovery situations.