Signal Strength Based Mote Localization

Advanced Topics in Wireless and Sensor Networks (CS 647)

Course Project

Ashutosh Dhekne (07305016)

Rahul Jain (07305022)

Department of Computer Science and Engineering,

IIT Bombay

Acknowledgements

We express our sincere thanks to Prof. Purushottam Kulkarni for helping us at every juncture in the project. His innovative ideas and our discussions with him always gave fruitful directions to our work.

We also thank the staff in the Computer Science and Engineering Department for providing us an uninterrupted access to our test bed at the oddest of hours.

Table of Contents

Table of Figures

List of Tables

1 Introduction

Many applications that employ moving motes require to know the precise location of the mobile node for their correct working. We envision a mechanism in which four fixed nodes will be present at predetermined locations and a mobile node will be able to localize itself using the variation in signal strengths received from these fixed nodes. Initially, the mobile node will move through the entire field collecting the signal strengths of all the four fixed nodes as seen form that location. This information will be stored in a database. When the mobile node is deployed in the field, it will request the fixed nodes to send packets to determine the signal strength form each of them. It will send this information to a base station where the database of the previously run training phase is kept. The base station will then infer the location of the mobile node.

2 Architecture

We use a structure as depicted in Figure 1. The mobile node asks for packets from the fixed nodes located at the corner of the grid. The packet signal strength collected by the mobile node is sent to the base station. Each of these components is a module and has its own separate program. The higher level overview of our project is given by Figure 2.

Communication between the base station, the mobile node and the fixed nodes happens through the standard Telos TOS_Msg structure packets. We have encoded the payload to suit our requirements so that we can encode the information regarding the coordinates of the fixed nodes, the signal strengths of the fixed nodes and the like. The payload structure for the communication packet from the base station to the mobile node is as follows:

SID	TID	Seq	X	Y

The SID field indicates the SourceID and the TID field indicates the TargetID. During the training phase, the X and Y fields are populated by the known X and Y locations of the mobile mote. The communication between the mobile node and the fixed node also happens through the same message payload structure.

When the mobile node sends data to the base station regarding the signal strengths of the various fixed nodes, it uses an entirely different message structure.

SID	TID	Seq	X	Y	FID	St1	St2	St3	St4

This structure includes the St1…4 fields that are used to store the signal strength value of the fixed node encoded in the FID field.

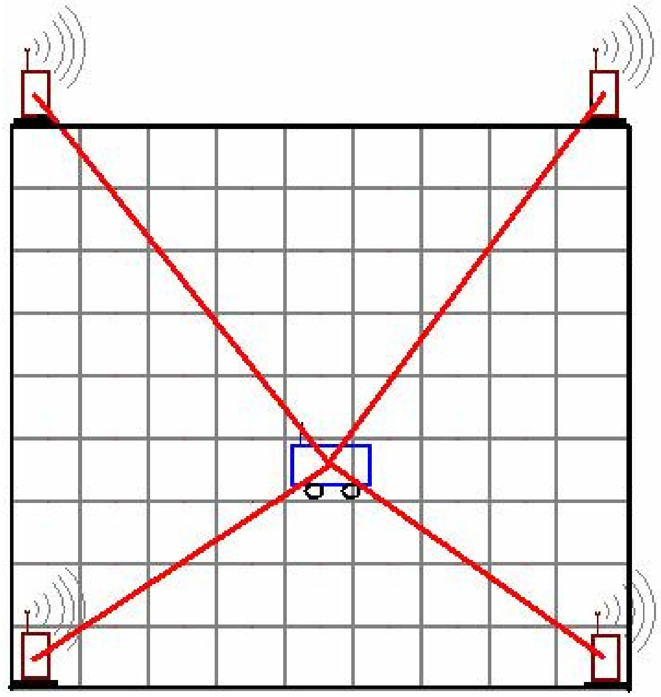

Figure 1 The grid setup

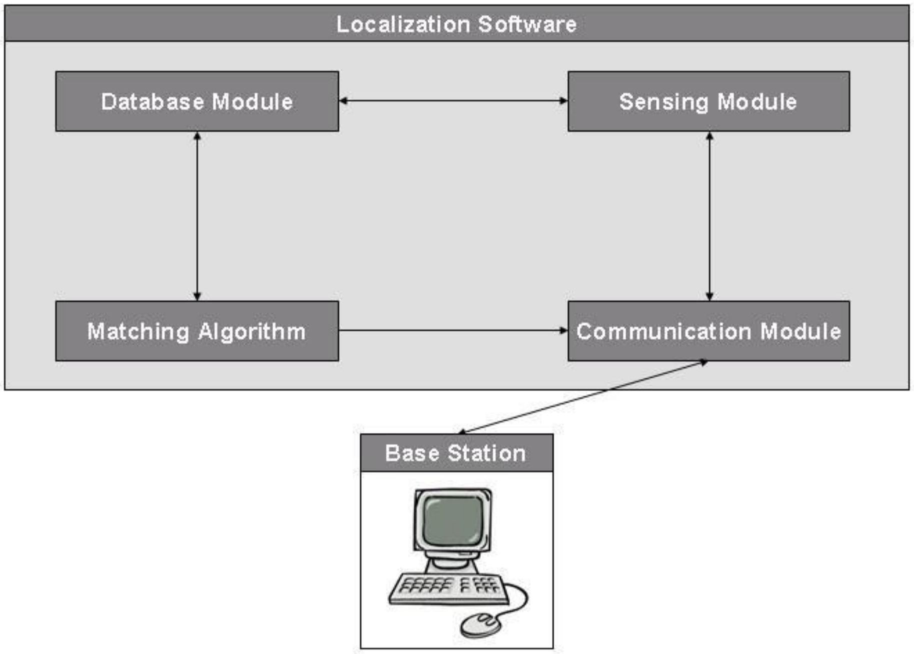

Figure 2 Modular design of the project

We shall now look at the different modules in detail.

2.1 Fixed Node

The fixed nodes just sit at their predefined locations and listen for requests to transmit packets. They receive requests to transmit data from the mobile node. This request is a special packet with a TargetID inside the payload of the packet indicating which fixed node must respond. The appropriate fixed node sends a stream of 16 packets to the mobile node, with the targetID set to the ID of the Mobile node and the NodeID feild set to its own ID. This allows the mobile node to ignore any packets received from other sources. The tinyOS code for the fixed node is the simplest. It contains a small task that sends 16 packets to the mobile node after hearing form it for a request to do so.

2.2 Mobile Node

The mobile node responds to a packet received from the base station, asking it to localize itself, by issuing requests to the four fixed nodes in sequence. The mobile node sends a special packet with the targetID set to the ID of the first fixed node and then waits for the 16 packets from the fixed node to be received. It stores the signal strength readings of these 16 packets in an array. It then sends the special packet to the next fixed node and so on. After it has collected all the packets, it sends the signal strength values to the base station. These readings are sent to the base station in a series of 16 packets, after being appropriately packed.

The tinyOS code on the mobile node is more complicated than that of the fixed node. The mobile node listens for a packet from the base station to begin its work. When the base station sends such a request, the mobile node changes state and sends a packet to the first fixed node. It then waits for 16 packets to be received. If it does not receive the said number of packets from the fixed node, it times out and retries. After correct receipt of all the 16 packets from the first node, a similar protocol is executed with the second fixed node and so on. The mobile node has a two dimensional array to store the 16 readings for each of the four receivers. After reception of every packet from a fixed node, this array is appropriately populated. Sending this data to the base station is slightly tricky since the two dimensional array is split up while sending it back to the base station. The mobile node sends in all 16 packets to the base station containing the 64 signal strength readings collected in all.

2.3 Base Station

Physically, the base station is a combination of the computer and a mote connected to it through USB. The mote connected to the computer sends a packet to the mobile node to start collecting the signal strength readings. It uses the TOSBase code for sending this packet. The program on the computer is a sophisticated version of the send program that presents TOSBase with an appropriate packet to send on air. In addition, it has a component to listen to the return messages from the mobile node, which contains the signal strength readings. The base station, however, does much more than just sends to and receives from the mobile node.

The base station code has a java user interface through which the user can issue appropriate commands to the mobile node. During the training phase, the user types in the X and the Y coordinates and then asks the mobile node to ``Collect Readings". All the reading collected, along with their X and Y coordinates are stored in an ArrayList that can be written to the disk as an object file. Once the training phase is completed, during deployment, the user may just “Query” the mote to initiate a similar data collection from the mobile node. The new data is then analyzed and matched with the data available from the training phase. The matching algorithms used are described in the next Section.

3 Algorithm

The matching algorithm needs to check the corellation between the readings collected during the training phase and those obtained in the deployment. We have used three algorithms to predict the location of the mobile node. As described in the Section, none of these algorithms gives us perfect results, and it seems trial and error is the only way to fix the usage of a particular algorithm.

3.1 SPM with expanding variance

In this algorithm, we check if all the signal strength values obtained during this test fall within the range of those obtained during the training. For the entire training data, we find the average variance of signal strength values. Signal strengths obtained in a new test phase is allowed a laxity within this average variance. The entire bunch of training data set is checked for possible matches. If more than one location shows a match, all these locations are returned. If only a single location shows a match, only this reading is returned, which is also the best result. If no readings match the new test reading, we allow an even more laxity of twice the variance and so on. We do not allow this to go on infinitely, and return zero results after a set degree of laxity.

3.2 SPM with percentage match

The idea used in this algorithm is similar to that used in the algorithm described in Section 3.1. Instead of increasing the variance, however, we allow lesser number of readings to match the training set. We expect better results through this algorithm, if few of the readings are excessively off from the general readings.

3.3 Norm2 method

We calculate the sum of squares of the difference between each reading of the test data with the corresponding reading in the training data for each entry in the training data set. The location, for which this second norm happens to be the least, is returned as the answer. This algorithm always returns only a single location since it checks for the least summation.

4 Experiments

The fundamental experiment in our project was to collect the readings for all the 167 points on the grid that we had plotted. After these readings were available, we plotted the signal strength with respect to the distance from each fixed node. We observed a large amount of variance in the signal strength values for the same distance. This indicates that it is difficult to predict the distance from a particular node just looking at its signal strength reading. This premise turned out to be true when we ran the test phase of our experiment.

The following figures show the signal strength variation with respect to distance.

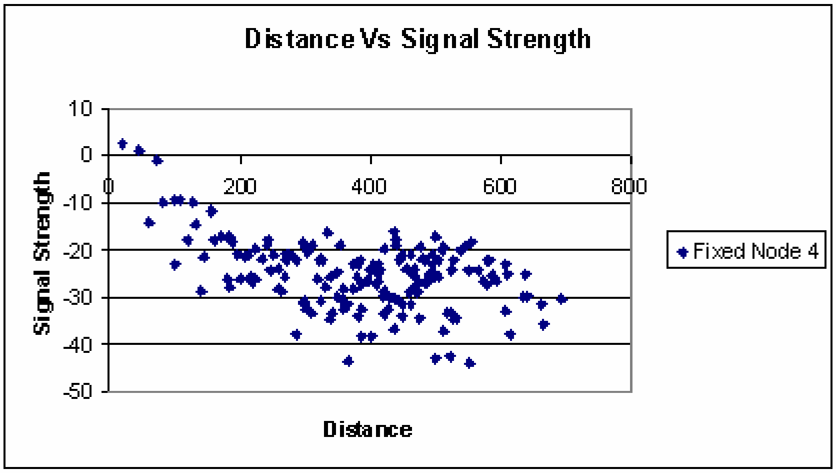

Figure 3 Signal Strength Variation with respect to Distance for Fixed Node1

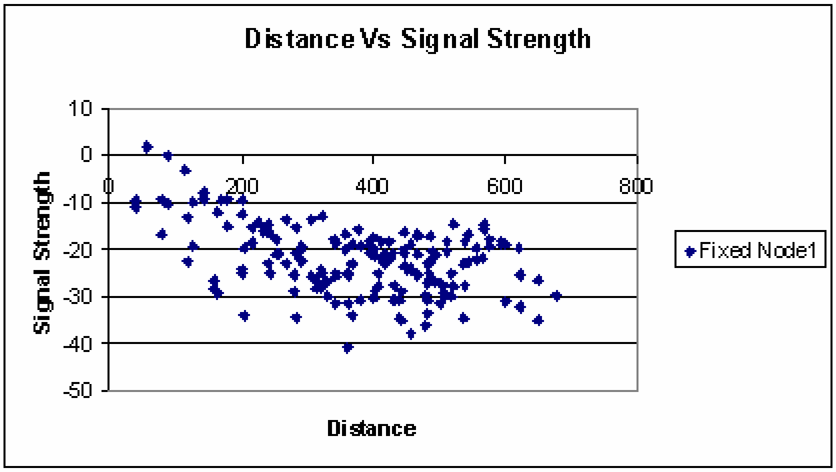

Figure 4 Signal Strength Variation with respect to Distance for Fixed Node2

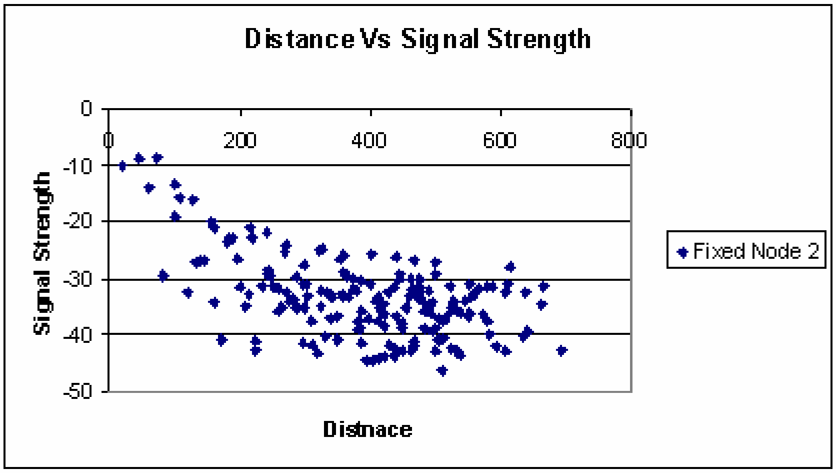

Figure 5 Signal Strength Variation with respect to Distance for Fixed Node3

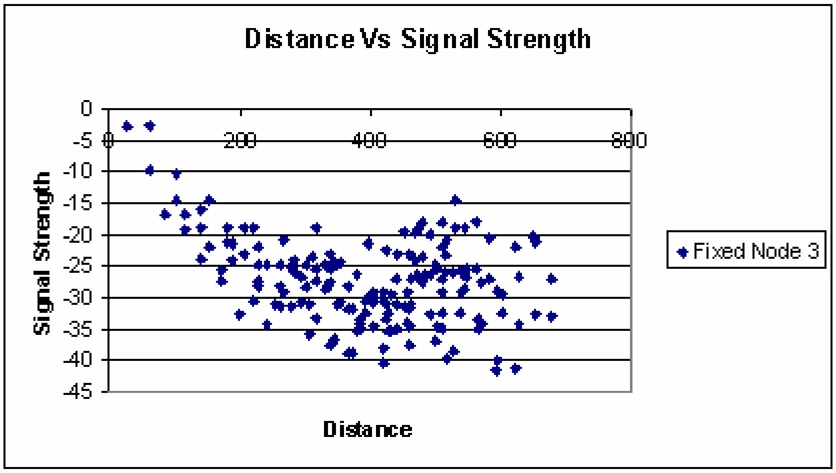

Figure 6 Signal Strength Variation with respect to Distance for Fixed Node4

The problem of localization was made very difficult by these variations in the training data itself. For good localization to be possible, we must have the temporal variation at least less than the spatial variation.

To profile how the signal strength actually varies in the indoor setting that we used, we perfomed a simple one shot experiment. We increased the distance between two consecutive readings and varied the transmission power.

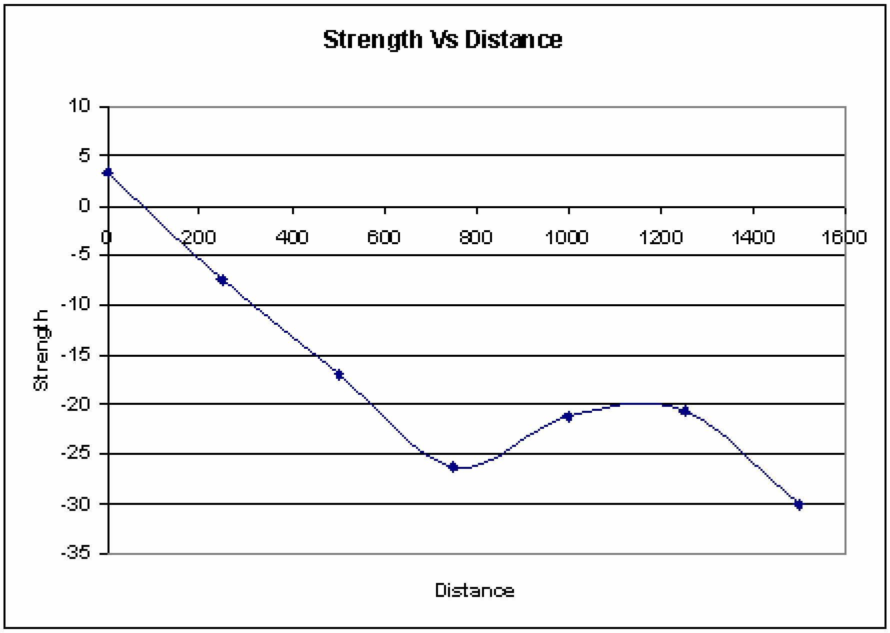

Figure 7 Signal Strength with Distance at Transmit Power 24

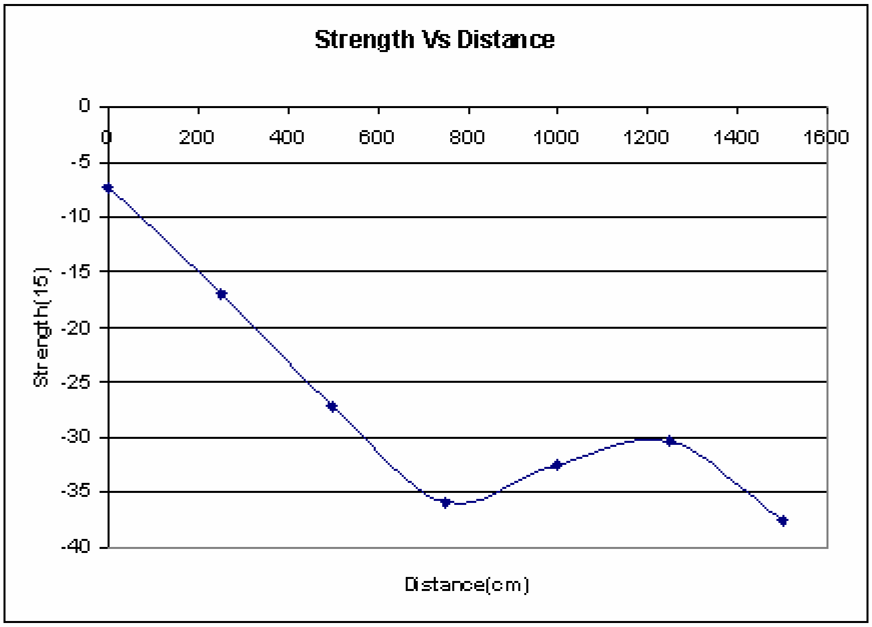

Figure 8 Signal Strength With Distance at Transmit Power 15

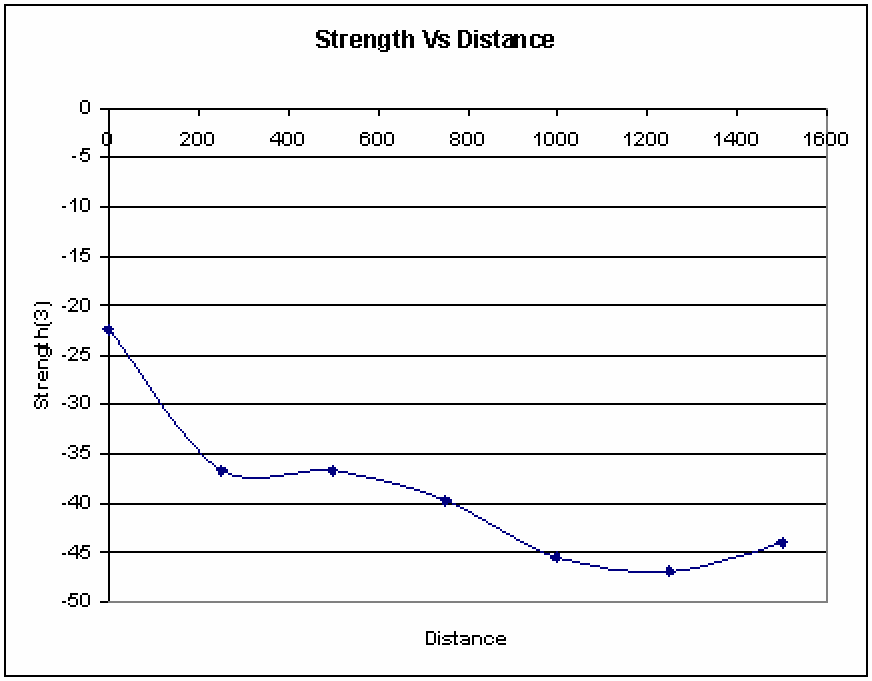

Figure 9 Signal Strength with Distance at Transmit Power 3

Going ahead with producing test data, we wanted to see if the test data taken few days after the initial training data was taken, also shows the signal strength variability that we observed in Figure 3 and others. Hence, we plotted a similar signal variation graph.

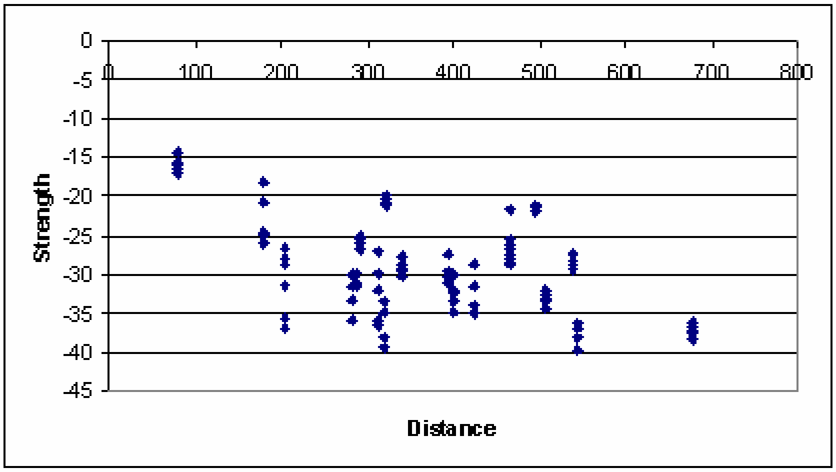

Figure 10 Signal Strength variations with Respect to Distance for Fixed Node 1

We observed similar readings for the three other fixed nodes but have deferred showing the graphs here for brevity.

We now were interested in finding out how much variation in the signal strength was tolerable by the algorithms. For this, we split up the 16 readings used in the training data into eight readings treated as training and other 8 readings treated as test data. In this case, we could get substantially better results. The following table summarizes our findings.

Table 1 Percentage Matching with a 50-50 split of the original training data

Percent Match	True match	Probable	%Match
100%	15	2	8.982036
95%	29	3	17.36527
90%	65	3	38.92216
80%	127	4	76.0479
75%	148	9	88.62275
60%	164	16	98.20359
55%	167	18	100

Since the intra-value variance of the original training readings is very small, we infer that the mechanism works reasonably well when the variance in the signal strength values is very less. This is indicated clearly in the Figure 11

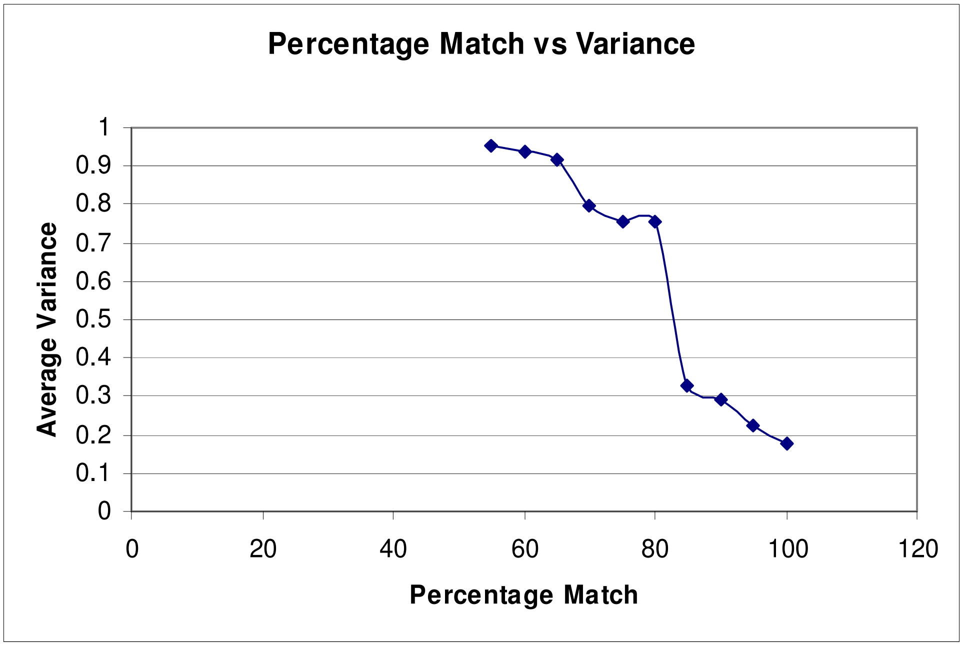

Figure 11 Tolerance of the Percent Match algorithm with respect to the variance in the readings

Since it is a standard mechanism to change the split size and check the results, we used the 75-25% split and observed that the initial matches were much more than that shown in the 50-50% split case. However, as the percentage match was allowed to decrease, both the 50-50% split and the 75-25% split converged with about the same performance.

Table 2 Percentage Matching by using the 75-25% split mechanism

PercentMatch	True match	Probable	%Match
100%	30	3	17.96407
95%	30	3	17.96407
90%	55	3	32.93413
85%	82	3	49.1018
80%	114	3	68.26347
75%	143	8	85.62874
70%	143	8	85.62874
65%	151	12	90.41916
60%	156	15	93.41317
55%	163	18	97.60479
50%	166	44	99.4012
45%	166	45	99.4012

When we use this approach, though the algorithm returns the correct matches, it also returns few locations that are wrong. The “Probable” column in the Table 1 and Table 2 indicates the extent to which the algorithm returned these wrong values.

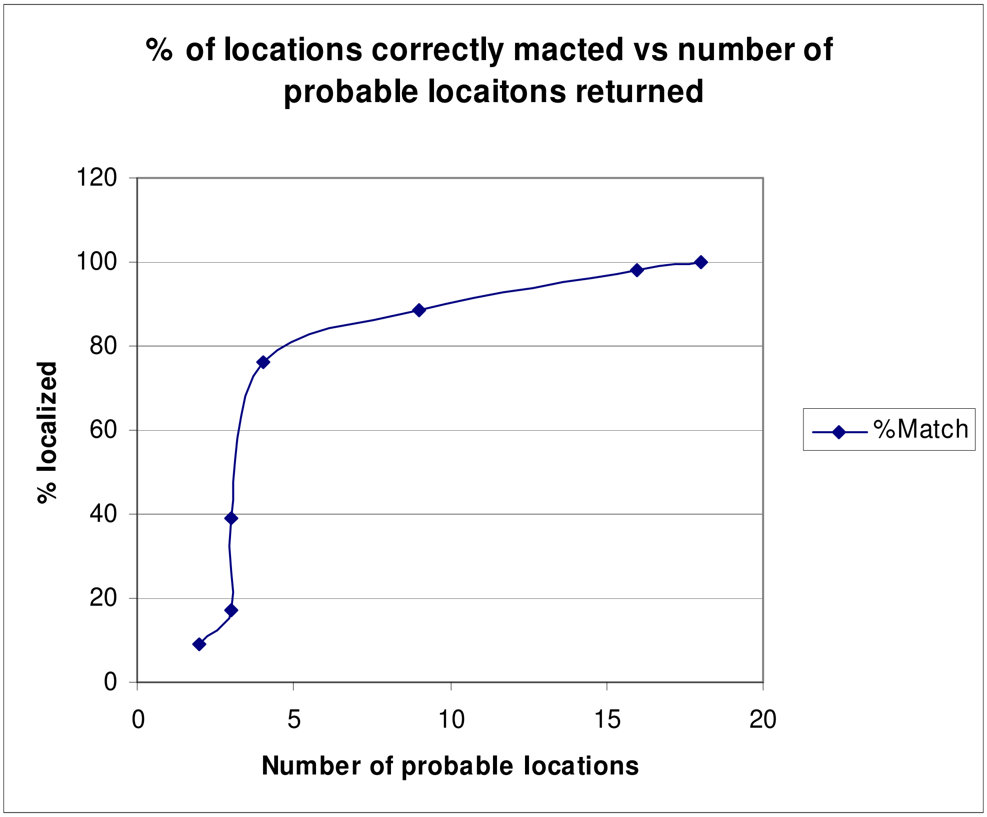

Figure 12 Percentage of grid points correctly localized vs Number of total locations returned (50-50% split)

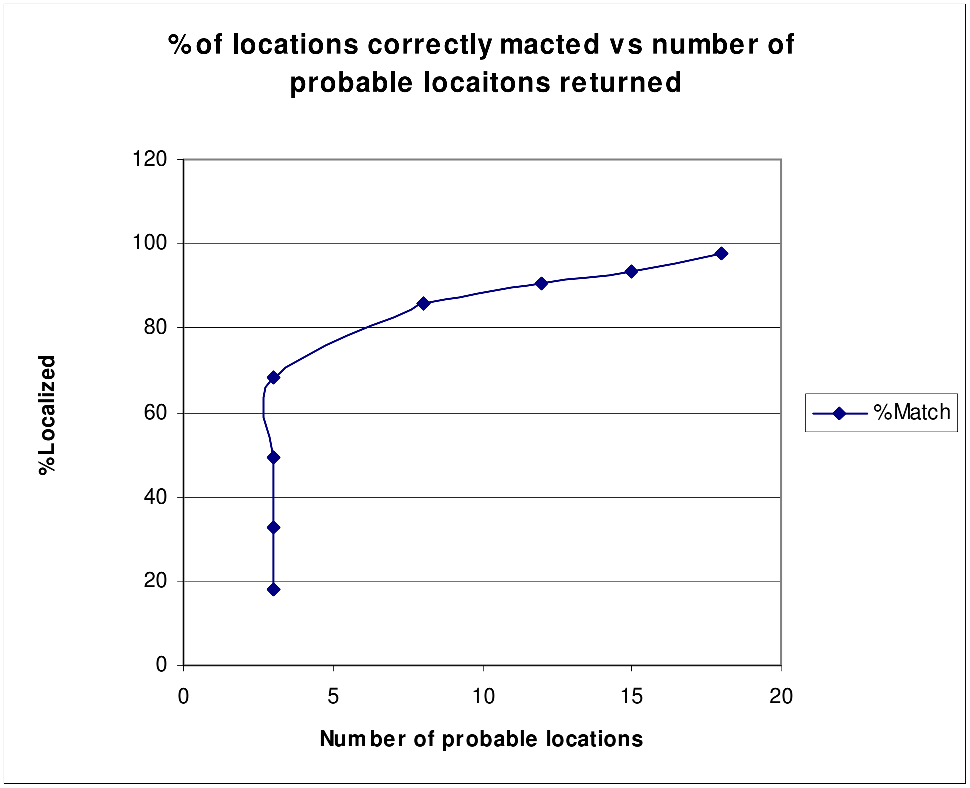

Figure 13 Percentage of grid points correctly localized vs Number of total locations returned (75-25% split)

We then went ahead and tried to localize twenty randomly selected points on the grid. As we expected, there was a substantial difference between the readings taken during the test phase and those taken during the training phase.

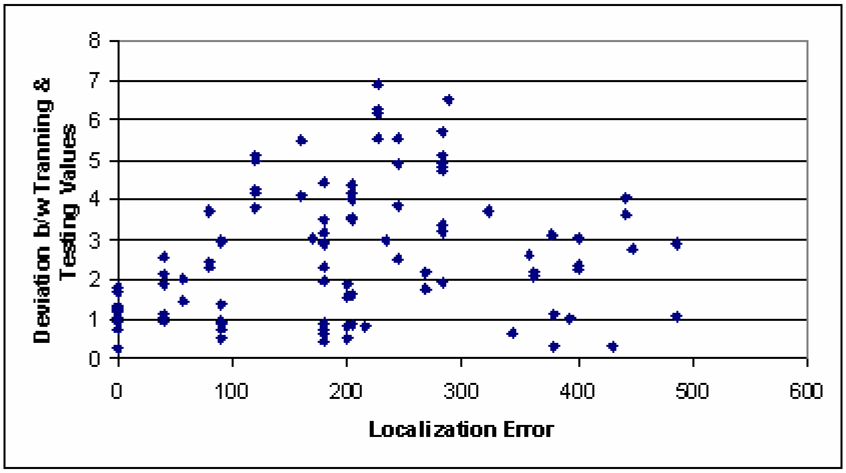

Figure 14 Deviation between Test values and Training values at the same point on the grid

As a result, the localization error that we incurred sorted in ascending order looked as shown in Figure 15 for the Norm2 algorithm.

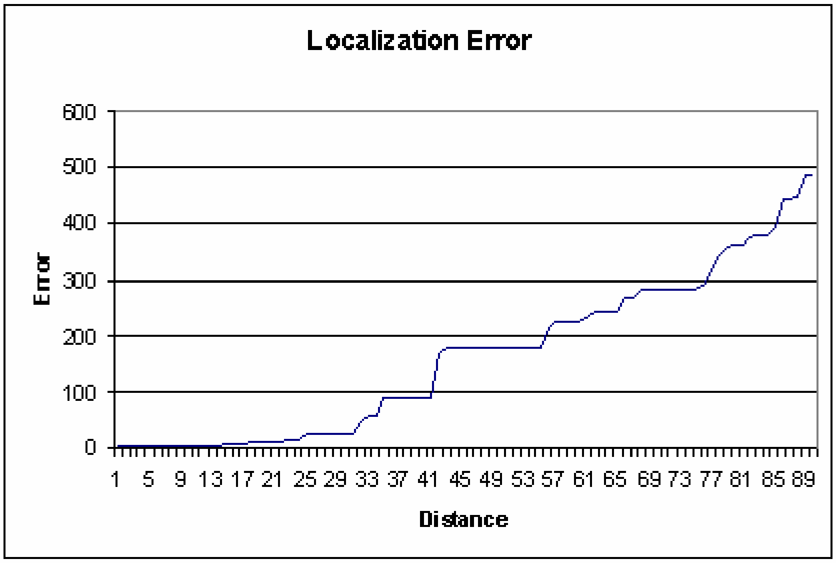

Figure 15 Localization Error sorted in ascending order

In general, for all the algorithms, the localization error was quite high. We plotted this error in three-dimensional graphs to find out if the localization error is clustered in particular parts of the grid. We observed that the localization error is almost uniform in all places on the grid. However, to our surprise, there were two locations where the Norm2 algorithm could return the correct location and incurred zero localization error. These two locations were placed at diagonal corners very close to one of the fixed nodes.

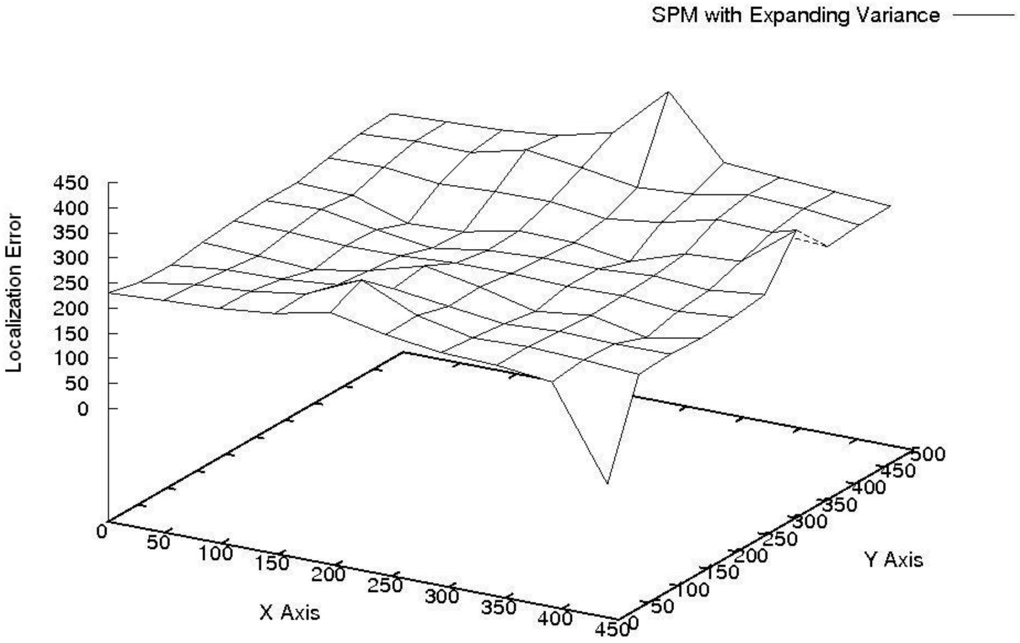

Figure 16 Localization errors at all parts of the grid (SPM with Expanding Variance)

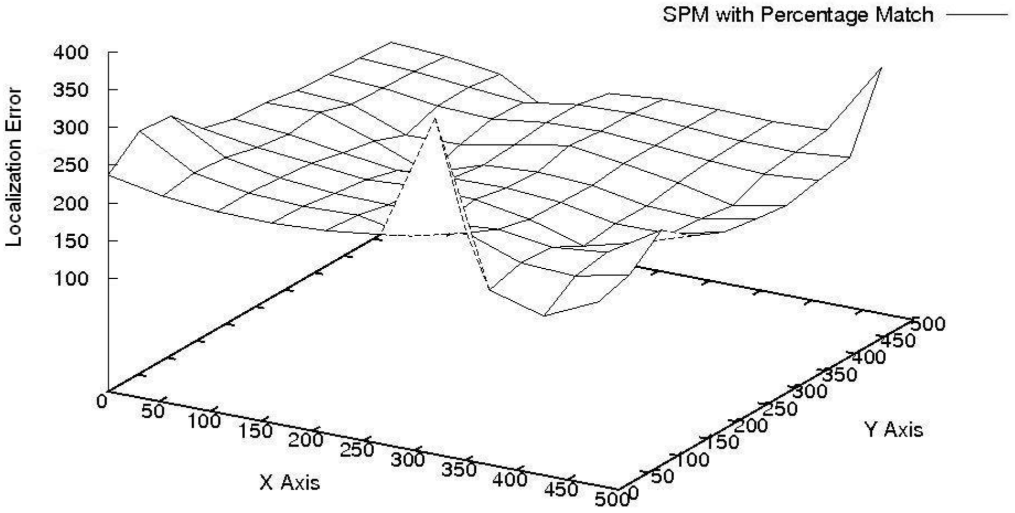

Figure 17 Localization errors at all point of the grid (SPM with Percentage Match)

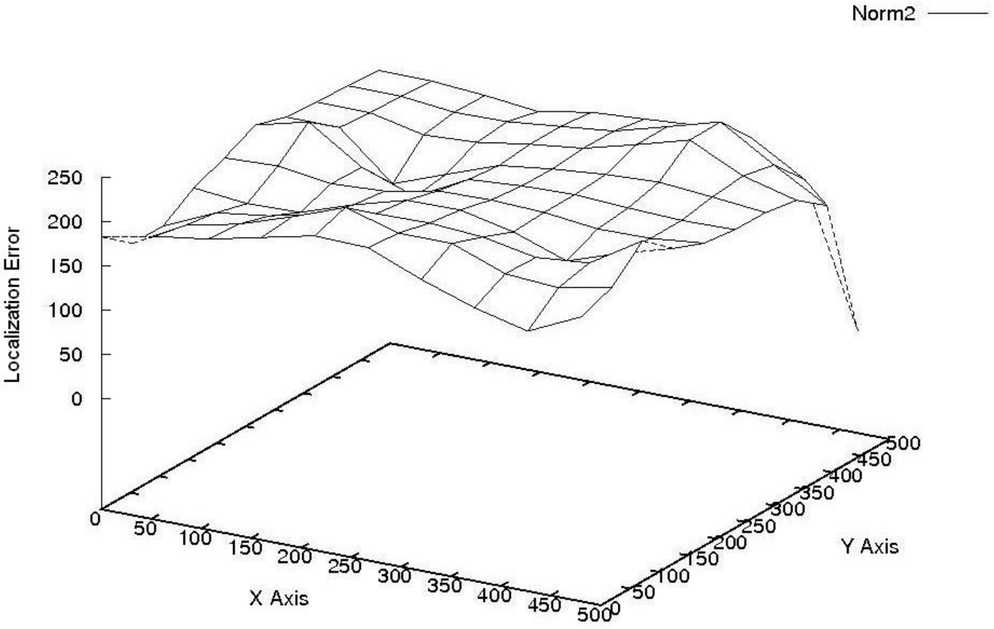

Figure 18 Localization errors at all points of the grid (Norm2 Algorithm)

5 Conclusion

It seems difficult to localize motes at the granularity we chose if the variance between the training and the testing phase is very high. However, if the variance can be kept low, we feel that signal strength based measures can still be used for localization. We performed all our experiments inside a large circular hall. We may have seen completely different observations had these experiments been done on a vacant ground. The circular hall may be causing multi-path fading due to its walls and roof.

In addition, we observed substantial external wireless activity inside the hall. If it was a source of interference in our readings, we could have two solutions. The first is to use different channel to communicate with our motes being careful not to use any of the channels that are being used in the vicinity by the access points. The second was to use a interference free area like large grounds. Decreasing the granularity of our measurements, that is, measuring every one-meter rather than every forty-centimeter as we did, might have resulted in better spatial correlation.

We need to remember that localization using the mechanisms discussed here is possible only if the spatial variance is larger than the temporal variance shown by the motes.

6 References

The Limits of Localization Using Signal Strength: A Comparative Study, Eiman Elnahrawy, Xiaoyan Li, and Richard P. Martin, *SECON 2004*

Tiny OS, http://tinyos.net/

Wikipedia. Variance. http://en.wikipedia.org/wiki/Variance

Gnuplot FAQ. http://www.gnuplot.info/faq/faq.html