LOCALE: Collaborative Localization Estimation for Sparse Mobile Sensor Networks

Pei Zhang and Margaret Martonosi
Department of Electrical Engineering
Princeton University
{peizhang, mrm}@princeton.edu

Abstract

As the field of sensor networks matures, research in this area is focusing not only on fixed networks, but also on mobile sensor networks. For many reasons, both technical and logistical, such networks will often be very sparse for all or part of their operation, sometimes functioning more as disruption-tolerant networks (DTNs). While much work has been done on localization methods for densely populated fixed networks, most of these methods are inefficient or ineffective for sparse mobile networks, where connections can be infrequent. While some mobile networks rely on fixed location beacons or per-node, onboard GPS, these methods are not always possible due to cost, power and other constraints.

In this paper we present the Low-density Collaborative Ad-Hoc Localization Estimation (LOCALE) system for sparse sensor networks. In LOCALE, each node estimates its own position, and collaboratively refines that location estimate by updating its prediction based on neighbors it encounters. Nodes also estimate (as a probability density function) the likelihood their prediction is accurate. We evaluate LOCALE's collaborative localization both through real implementations running on sensor nodes, as well as through simulations of larger systems. We consider scenarios of varying density (down to 0.02 neighbors per communication attempt), as well as scenarios that demonstrate LOCALE's resilience in the face of extremely-inaccurate individual nodes. Overall, our algorithms yield up to a median of 21X better accuracy for location estimation compared to existing approaches. In addition, by allowing nodes to refine location estimates collaboratively, LOCALE also reduces the need for fixed location beacons (i.e. GPS-enabled beacon towers) by as much as 64X.

1 Introduction

As sensor network deployments become reality, sensor network applications have become more diverse in form and function. Many applications involve mobile networks that are made up of nodes with unpredictable movement patterns [14][16]. Because of these unpredictable movements, there will be sparse areas in the network. Furthermore, other factors, such as cost, will further limit network node densities. In these sparse areas the networks need to behave as disruption-tolerant networks (DTNs). Yet it is common, in these sparse mobile sensor networks, to require up-to-date location information for each node in the network [17][21].

One way to localize sparse mobile nodes is by using a Global Positioning System (GPS) on each node. Unfortunately, many prerequisites have to be met for proper GPS function. The GPS antenna must have a clear view of the sky, making it difficult for use indoors or in urban canyons. Furthermore, the power consumption of such devices greatly shortens the lifetime of the sensor nodes, and greatly increases the cost of each node.

To solve these problems, several mobile networks use location beacons as localization references. Nodes use their proximity to these fixed or mobile beacons to estimate their own locations [24]. However, this method requires the beacons to cover all areas of the network where localization is desired; this can translate into high infrastructure costs.

To reduce the infrastructure requirements in dense networks, many collaborative methods have been developed. Typically, nodes in these networks localize by collaboratively deducing network topology and using several anchor beacons to calculate absolute locations [19]. If nodes are mobile, however, these methods become inefficient or ineffective due to increasing collaboration overhead. Furthermore, in sparse mobile networks, naive collaboration becomes impossible due to lack of nodes in range.

In this paper, we present Low-density Collaborative Ad-Hoc Localization Estimation (LOCALE). Our method is a distributed localization algorithm designed to enable collaborative localization in sparse mobile sensor networks. It not only merges location information when neighbors are present, but also actively predicts and maintains the location estimation during periods of disconnection.

LOCALE maintains an ongoing, rough estimation of the nodes' location and certainty during disconnects by using a dead-reckoning (DR) system. (This is simpler, cheaper, and more energy-efficient than per-node GPS.) When nodes meet a neighbor, they swap position estimates and then refine the nodes' location by a linear combination of the two estimates, weighted by the variances. Over time, in a delay tolerant manner, LOCALE effectively averages movement of nodes and gives each node a distribution describing its lo-

978-0-7695-3157-1/08 $25.00 © 2008 IEEE
DOI 10.1109/IPSN.2008.63

cation. With such distributions the nodes maintain not only a prediction of their actual location, but also a "confidence estimate" of the likely accuracy of this prediction.

LOCALE has the following key characteristics:

- It maintains a node's location estimation with movement tracking when neighbors are not present. It refines the location estimation with information swapping when neighbors are encountered.
- Provides accurate location information with median of up to 21X less area error compared to the commonly used beacon-tracking method.
- Reduces infrastructure requirements by up to 64X, while maintaining location accuracy.
- Offers more than 10X faster error correction for nodes with inaccurate estimations.
- Provides broad applicability to sparse, dense and heterogeneous systems without modification.
- Dissipate to 150X less power than per-node GPS.

In this paper, we evaluate our collaborative method against a baseline method that acquires localization only directly from location beacons and maintain the estimation with a dead-reckoning system. LOCALE, when compared with the baseline method, produces a much more accurate measurement with 75th-percentile area error reduced by up to 27X. In addition, LOCALE is resilient to the case where nodes with large location errors are introduced into the network. Through its collaborative approach, these large errors can be quickly reduced. This method allows the entire sparse sensor network to operate with much less beacon infrastructure (64X reduction), while maintaining accuracy.

This paper is organized as follows. Section 2 provides the details of our system. Section 3 shows measured results in a real sensor node implementation. Section 4 shows large scale simulated results based on the measured parameters. Section 5 discusses related work. Section 6 gives the conclusions.

2 Collaborative Location Estimation

Low-density Collaborative Ad-hoc Location Estimation (LOCALE) is a distributed collaborative localization algorithm designed to solve the localization problem in highly partitioned, sparse mobile sensor networks. LOCALE not only maintains a location estimate, but also a "confidence cloud" indicating the likelihood of that estimate's accuracy. Most importantly, LOCALE nodes actively refine the location estimate when neighboring nodes come into communication range.

LOCALE features three major phases, shown in Figure 1, to maintain and refine location estimations. Section 2.2 describes the *local phase*, which uses the node's movement tracking information to maintain a cheap but possibly inaccurate location estimation. This phase allows the node to maintain location information during long periods of disconnection, but is not sufficiently accurate to be used by itself. Section 2.3 describes the *transform phase*, where the

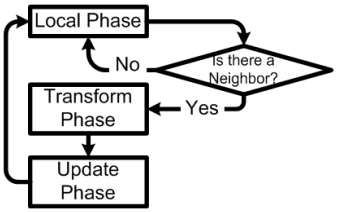

Figure 1. LOCALE overview: Location error increases during the local phase and decreases with collaboration in the update phase.

neighbor's location estimation is used to create an estimation of self location. Section 2.4 describes the *update phase*, where the estimation obtained from the neighbor and the existing estimation are combined. This phase allows accurately refining of the location estimate from different nodes.

The novelty of LOCALE is that it is, to the best of our knowledge, the first delay-tolerant, collaborative localization policy that is effective for sparse mobile sensor networks.

Before describing LOCALE's three phases in detail, we first describe how LOCALE represents a node's position.

2.1 Location in LOCALE

In order for LOCALE to predict and merge localization information from multiple estimations, it requires not only the location estimation, but also the certainty of the estimation. Since, by using LOCALE, each node's location is in essence an average of location estimations of nodes in the network, we use a normal distribution to represent the location estimation (mean) and the certainty (variance). While a single node's location estimation may not follow a normal distribution, by the Central Limit Theorem, the averaged estimation should approach a normal distribution. This assumption allows the use of well established distribution merging methods described in Section 2.4. In Section 4.6, we also explore the effect on LOCALE's performance when this assumption does not hold, i.e. the movement model is highly non-normal.

To represent the distribution we use the probability density function:

$$p(X) = \frac{1}{2\pi\sqrt{|C|}} * e^{-\frac{1}{2}(X-\bar{X})^T C^{-1}(X-\bar{X})} \quad (1)$$

The equation represents the probability of the true location for node (X) relative to the estimated location ($\bar{X}$). To define the equation we need only the estimated location ($\bar{X}$) and the covariance matrix C. For simplification purposes, we consider the two-dimensional case in this paper; however this work can be easily extended to three-dimensions

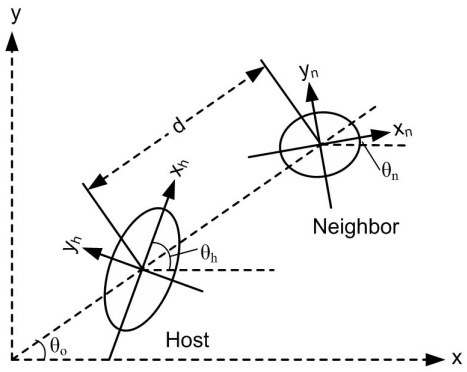

Figure 2. Representation of 2 neighboring nodes with different orientations.

with altitude information.

$$C = \begin{pmatrix} \sigma_x^2 & \rho\sigma_x\sigma_y \\ \rho\sigma_x\sigma_y & \sigma_y^2 \end{pmatrix} \qquad X = \begin{pmatrix} x \\ y \end{pmatrix} \tag{2}$$

The main diagonal of the matrix are the variances along the axes of its coordinate system, and the other values are the covariance between the two axes. These two variables are updated by LOCALE as the nodes move and meet neighbors. Initially, they should be initialized with the start location, this can be accomplished by deploying near a location beacon.

To define the distribution and its orientation, LOCALE keeps 3 variables: the location estimation $\bar{X}$, the covariance matrix C, and the angle θ between the local coordinates and the global coordinates. Figure 2 illustrates our definitions of the relative angle of the neighboring nodes θ_o, where each node has its own local coordinate $(x_h, y_h)(x_n, y_n)$, and an angle (θ_h, θ_n) relating it to the global coordinate (x, y).

We maintain an estimation of the node's location approximated using this multi-dimensional normal distribution, allowing LOCALE to estimate both location and confidence in terms of variance. The variance allows the application or the end user to calculate the confidence interval and interpret the location estimation accordingly. Furthermore, when nodes are expressed using this method, information from systems with heterogeneous hardware can also be incorporated. Finally, we note that while this representation (and the math to manipulate it) may appear complex, Section 3 describes its implementation on an MSP 430 embedded processor which is used in several sensor nodes.

2.2 Local Phase

To maintain the location estimation of a moving node through long periods of disconnection, a local phase is needed. In LOCALE, each node maintains a local position estimation based on one of a variety of simple, existing, low-cost, low-energy movement tracking methods.

While some nodes have onboard location sensors, such as GPS sensors, LOCALE strives to reduce the number of nodes with such expensive sensors. With LOCALE, most nodes can use cheaper, low-accuracy dead-reckoning sensors to track their movement relative to their last measured location [43]. These sensors are currently available on some sensor nodes [5], as well as other more capable devices such as cell phones or laptops.

LOCALE incorporates measurements from the movement tracking algorithm After each step in the local phase, the new distribution is then the combination of relative measurement distribution and the existing estimation distribution.

$$N = N_{old}(X_1, C_1) + N_{delta}(X_2, C_2) \tag{3}$$

Together this gives the new distribution with mean and variance as

$$N_{combined} = N(X_1 + X_2, C_1 + C_2) \tag{4}$$

To incorporate the relative measurement distribution, both must be in the same global coordinate system. However, the movement covariance matrix is oriented in the direction moved. The covariance matrix in local coordinate C_L is described as:

$$C_L = \begin{pmatrix} \sigma_{x'}^2 & 0 \\ 0 & \sigma_{y'}^2 \end{pmatrix} \tag{5}$$

The local covariance matrix is rotated to the global coordinate by

$$C = R(-\theta)^T C_L R(-\theta) \tag{6}$$

Where θ is the direction the node moved, and the rotational matrix is defined as

$$R(\theta) = \begin{pmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{pmatrix} \tag{7}$$

Finally, the mean and covariance matrix of the new estimated location distribution is simply the summation.

The novelty of incorporating the relative movement tracking information lies in the ability of the system to merge the movement information along with the covariance information. Thus movement tracking generates not only information on the estimated location, but also the new estimate of accuracy.

2.3 Transform Phase

LOCALE merges location information of the nodes in order to create a more accurate estimation than one node can achieve alone. Since neighboring nodes are not at the same location, the distance between the nodes prevents a node from simply copying a confident neighbor's position estimate and merging it with its own. Instead, to transform the neighbor's estimate so that it is suitable for merging, we must transform it to an observation on the host's location. In order to achieve this, we need to obtain information on the relative location between the two nodes. Figure 4 illustrates the transform process, where neighbor transformation is shown on the right.

LOCALE supports different forms and quality levels of relative node-to-node location information, including both relative measurement of range and direction of the

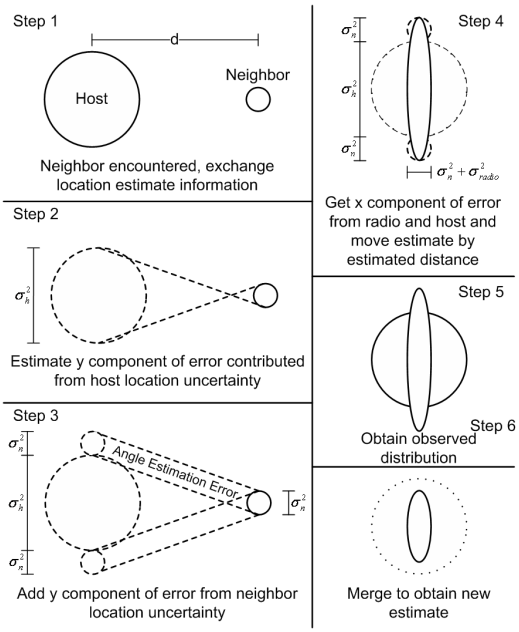

Figure 3. Relative location estimation with no direction measurement.

neighbor, relative range to the neighbor, or only very simple information indicating that the neighbor is somewhere within communication range of the node. There are various methods to obtain both range and direction information [18][20][26][32]. Since most sensor nodes do not have very specialized hardware, however, this paper only assumes the simplest case. Namely, we assume that nodes are only able to discern that another neighbor is within communication range, with a distribution describing the nodes' distance while they are in range.

While formal methods exist to account for this information, we present and use a heuristic method that reduces complex matrix manipulation and relies mostly on inner matrix additions [15]. This method is easily implemented on low-power micro-controllers. Figure 3 shows the conceptual steps performed to incorporate the neighbor's observation into the host's local frame. In Step 1, we obtain information from the location estimation and rotate the observations to comply with the relative coordinate, where the X-axis of the two observations coincide

$$C_h = R(\theta_o - \theta_h)^T C_{L_h} R(\theta_o - \theta_h) \quad (8)$$

$$C_n = R(\theta_o - \theta_n)^T C_{L_n} R(\theta_o - \theta_n) \quad (9)$$

In Step 2, the y-component of the transformed covariance matrix is calculated from the angle uncertainty caused by the host's location uncertainty. Step 3 adds in the additional contributions caused by the neighbor's location uncertainty. In Step 4, the transformed observation distribution is created by including the x component of the covariance matrix taking into account of variability of distance when nodes are in range, which is the sum of x component of the host variance and $Range^2(1 - 2\sqrt{2}/3)$. Since all nodes are oriented in the relative coordinates, the covariance of the error observations are 0. The mean of the observation to be merged is moved by distance d, the expected vector of distance between the two neighboring nodes when in radio range, which is $Range/\sqrt{2}$, in the direction $(\theta_o - \theta_n)$.

$$C_{Lobserved} = \begin{pmatrix} \sigma_{radio} + \sigma_n & 0 \\ 0 & \sigma_h + 2\sigma_n \end{pmatrix} \quad (10)$$

$$X_{observed} = \begin{pmatrix} x_n + d * cos(\theta_o - \theta_n) \\ y_n + d * sin(\theta_o - \theta_n) \end{pmatrix} \quad (11)$$

Finally, in Step 5, we rotate the transformed observation and the host distribution to the global coordinate system for the final merging in Step 6.

$$C_{observed} = R(-\theta_o)^T C_{Lobserved} R(-\theta_o) \quad (12)$$

The "in-range" distribution we rely on for neighbors, in general, gives the node a better estimate in the direction facing the neighbor node. However, as nodes move, they will encounter this or other neighbors in different orientations, and thus obtain a more accurate estimation in all directions. This in effect creates delayed triangulation even with only one neighboring mobile node.

The novelty of the transformation phase is how the system projects neighbor observation to a self observation, allowing for more accurate merging in the next phase. Furthermore, this method does not assume a particular radio profile or special hardware and only requires probabilistic measurements.

2.4 Update Phase

LOCALE improves node localization accuracy by merging observation between neighbors, when one is encountered. This essentially increases the number of observations on the node's location and averages out measurement errors. Due to different movement patterns, or measuring devices, each node has estimations with different certainties. Therefore, we combine the estimations weighted by their respective certainties, which are represented by their variances.

The distributions are merged as a weighted linear combination. Our matrix merging methodologies are inspired by prior robotics work by Smith and Cheeseman [35]. This section describes the self-estimation preparation and the final merging process shown in Figure 4. From the self-estimation (Section 2.2) and the neighbor observation (Section 2.3), we have two distributions that can be merged to create the new estimation. Due to the increased number of observations, we calculate the combination of these distributions as the harmonic mean. First we calculate the merge factor defined as:

$$K = C_h * [C_h + C_{observed}]^{-1} \quad (13)$$

The merge factor represents the weight each distribution has

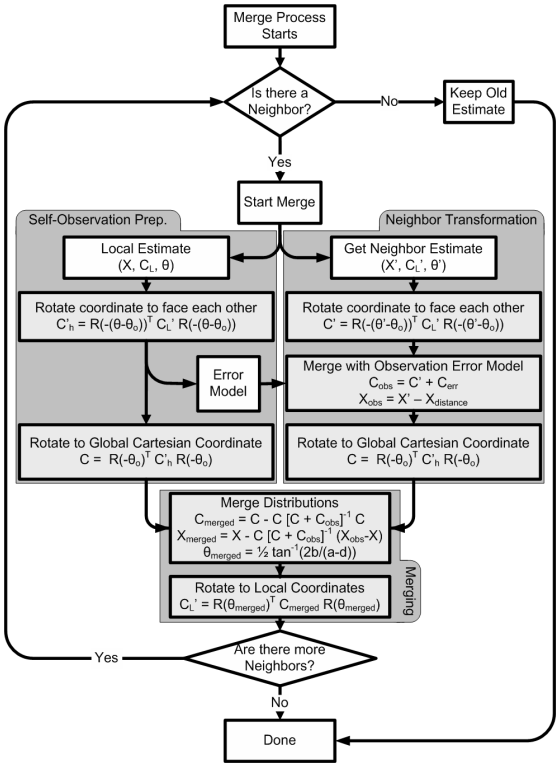

Figure 4. Block diagram of the merging process when neighbor is encountered.

on the result. This is used to calculate both the new covariance matrix and the new location estimation.

$$C_{merged} = C_h - KC_h \tag{14}$$

$$\bar{X}_{merged} = \bar{X}_h + K(\bar{X}_{observed} - \bar{X}_h) \tag{15}$$

We next obtain the new angle of the covariance matrix with

$$\theta = \frac{1}{2} tan^{-1}\left(\frac{2b}{a-d}\right) \qquad C = \begin{pmatrix} a & b \\ b & d \end{pmatrix} \tag{16}$$

and finally rotate the merged distribution back to the local coordinate.

$$C_{Lnew} = R(-\theta_{merged})^T C_{merged} R(-\theta_h) \tag{17}$$

The merged location and the covariance matrix are stored as the new self location estimate. Since the merging algorithm is a linear combination, the process can be repeated if more neighbors are in range.

This phase, in conjunction with other parts of LOCALE, enables delay-tolerant collaborative localization in extremely sparse networks.

2.5 Assumption of Independence

LOCALE assumes that all observations merged are approximately independent. If information flow in the system has loops (i.e. nodes repeatedly combine with each other without movement) the estimation could be erroneously "certain". Fortunately this is not a common problem in sparse mobile networks where both a) movement estimation between merges and b) time between connections, reduce the dependence between previously merged information. However, to mitigate problems when two nodes stay in range for long periods of time, we include a merge history table to record nodes' merging history. After two nodes merge 4 consecutive times, it will set a timer to prevent further repeated merges. This further decreases the likelihood of undesired merges of non-independent observations. We evaluate such situations with a strongly correlated movement model in Section 4.6.

3 Hardware Implementation

We designed LOCALE to function on hardware with a wide range of compute, storage, communication, and localization capabilities. As a proof of concept for LOCALE in real hardware, we implemented a two-dimensional LOCALE system on off-the-shelf sensor nodes. This prototype also provides measured system parameters for the large-scale simulations in Section 4.

The sensor network nodes used were the ZebraNet 5.2 test nodes. These nodes are electronically the same as nodes deployed for wildlife tracking in Kenya during the summer of 2005 [45]. The test node consists of several different peripherals. Of particular importance to LOCALE are the microcontroller, the radio, and the accelerometers. All ZebraNet nodes are equipped with GPS, but for LOCALE, the GPS is turned off on all nodes except the one being used as a beacon. Below, we discuss LOCALE's movement tracking, overhead, power consumption and real-world experiments of our hardware implementation.

3.1 Movement Tracking

The local phase of LOCALE requires nodes to track their movement. Since each ZebraNet node has a three-dimensional accelerometer [37], we use this to track its movement. To determine the orientation of the node, a motion sensor [33] or an electronic compass [30] module could be added. However, we only use the accelerometer as an implementation to demonstrate the basic concept of movement tracking for LOCALE.

We created a rudimentary tracking algorithm in order to test LOCALE. Tracking is performed by recording the accelerometer data and converting the acceleration information based on the equations:

$$d_{new} = \frac{1}{2} * a * t^2 + v_o * t + d_o \qquad v_{new} = a * t + v_o \tag{18}$$

The quantities v and d are calculated each time period, v

Function	Clock Cycles	Time
Prediction Merging	1,900,000	475ms
Movement incorporation	760,000	190ms

Table 1. LOCALE Merging Function Runtime, showing speed is acceptable even when implemented on low-end microcontrollers

is reset to 0 when no vibration is sensed and d is reset after each merge of LOCALE. Experiments were performed to determine the characteristics of this algorithm. Tracking errors were mostly within 15% of the distances moved in both X and Y direction. These measured parameters are collected and used for the simulations in the Section 4.

3.2 LOCALE Code

The ZebraNet nodes use the 16-bit MSP430F1611 microcontroller running at 4MHz. This is a popular processor that is also used in other sensor nodes [10]. Our demonstration that LOCALE can run on this platform indicates its practicality in similar systems such as MICA motes [8][9], as well as more capable systems such the Imote2[7], or the Stargate [6].

Code Size: LOCALE is designed for sensor network nodes, which mostly have strict memory constraints. Therefore, code memory usage needs to be kept low. While LOCALE uses floating point numbers to keep track of location and uses math functions for merging, the MSP430 processor is a integer processor. Thus, floating point operations, trig functions, division and multiplication are performed by software library functions provided by the MSP-GCC compiler. Even so, the total LOCALE code size including math functions, prediction, update and movement tracking codes, is only 17KB, corresponding to 35% of the available code flash. RAM memory usage is also low at less than 1.4KB, or 14% of total RAM.

Code Overhead: LOCALE must execute within a reasonable time as not to interfere with normal operation of the nodes. Table 1 shows the measured runtime of the main parts of LOCALE. In the table, prediction merging consists of both the transform phase and the update phase of LOCALE, whereas movement incorporation consists of the local phase, which incorporates the information gathered from the accelerometers. Even on the MSP430 processor running at 4MHz, LOCALE executes fairly quickly. The process speeds are especially acceptable considering the long periods of idleness in sparse sensor networks, for which LOCALE is designed. The entire process takes only around 650ms to complete. Since prediction merging only runs once every communication cycle, the impact on system performance is small. Furthermore, since these functions are not time-critical, the process can be scheduled to run only during idle time.

3.3 LOCALE Power Consumption

Most sensor nodes have very limited energy sources, so LOCALE's energy efficiency is extremely important.

	Power Consumption
LOCALE	430μW
GPS one sample per 8 minutes	4100μW
GPS one sample per second	66000μW

Table 2. LOCALE Power Comparison

For LOCALE, energy consumption stems mostly from two components: node movement tracking, and the communication needed to exchange location information with their neighbors. Here, we present the power consumption of our implementation, and compare it to the per-node GPS method.

Movement Tracking: For the motion tracking in the dead-reckoning "local phase", the accelerometer needs to record data periodically to avoid large sampling errors, with the sample frequency dependent on movement of the node. From our experiments, with our rudimentary tracking method, 32 samples per second is sufficient. Table 2 compares the average system power of LOCALE to ZebraNet nodes fitted with the Xemics GPS module [44] for localization. The Xemics RGPSM002 is an ultra low power GPS receiver module. From the table we see the constant measurements of the full-degree of movement would require only 430μW. Compared to the energy consumption for GPS receiver sampling every 8 minutes, this is a 9.5X reduction. Furthermore, when applications require location information even more often, LOCALE consumes 150X less power than obtaining one GPS sample per second.

Communications: The radio used on the ZebraNet node is the XTend^{TM} OEM RF Module [22]. The radio has a maximum transmit power of 1W, which gives roughly 2 km of outdoor range. Every two minutes, a radio communications cycle runs, in which nodes first send out peer discovery packets to discern if other nodes are within range. If so, the communication cycle can continue for up to 1 minute in order for nodes to exchange position data. During each communication cycle, the node's self estimation needs to be communicated to its neighbors. This consists of five 4-byte numbers totaling 20 bytes: X location, Y location, X variance, Y variance, and angle to the global coordinates. This translates to less than 5mJ per communication, or an average of 1.4μW of transmission overhead. However, since we piggyback this information into the previously empty peer discovery packets, this overhead is avoided in our implementation.

3.4 Real World Experiments

Small scale case study experiments were performed on vehicles as a proof-of-concept for LOCALE in real systems. While we show results from large scale simulations of LOCALE in Section 4, these case study experiments show the feasibility of the system.

In these tests, a node is placed on a car and travels approximately 300 meters west down the road at speeds of less than 15 miles per hour to point A where it is held stationary. Another node is placed in a car and travels 300 meter east to

Percentiles	25	50	75
Without Merge	26m	48m	90m
LOCALE	3.9m	21m	37m

Table 3. Vehicle case studies

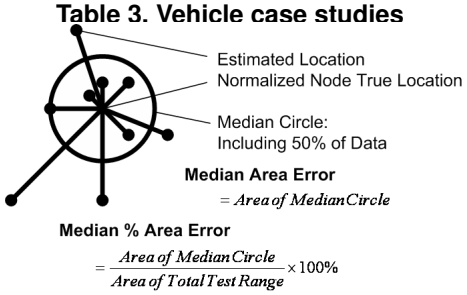

Figure 5. Definition of percent area error, a metric used in our simulations.

point A, where these nodes now exchange and merge their location estimations. This process was repeated 10 times. The radio was tuned down to 10mW power with a range of approximately 20 meters. The results of the experiment are shown in Table 3. We see that for all percentiles LOCALE performed significantly better than with movement tracking alone.

4 Simulation

To evaluate the performance of LOCALE in large-scale environments, we performed simulations based on parameters measured from the ZebraNet node implementation described in Section 3. In this section, we first explain our measurement metric, then evaluate LOCALE's performance in a small scale example, followed by evaluating various aspects of its performance in sparse sensor networks.

In our simulations, we measure the error as a vector that points from the estimated location to the node's true location. To show typical error, we use the median, 25 percentile and 75 percentile because they are less affected by abnormally large errors. However, because the error vectors point in multiple directions, we use the **median percent area error** to display the errors of multiple nodes. As shown in Figure 5, the median area error is the area of the smallest circle that includes 50% of the error vectors when their starting points are placed at the origin of the circle. The *median percent area error* is calculated by dividing the area error by the testing area. For example, when there is a median percent area error of 3%, it means that 50% of the estimations fall within a circle with 3% the area of the entire testing area. The 25 and 75 percent area error are the smallest circles that include 25% and 75% of the error vectors respectively. This metric gives an unit-less representation of error for multiple nodes that is irrespective of the size of the testing field.

Our simulations are performed in networks with densities shown in Table 4. In these simulations, we compare LOCALE with the baseline technique of the beacons method combined with dead-reckoning (DR) system. The

Number of Nodes	Movement Model	Neighbors
10	Random	0.02
100	Random	0.24
100	Zebra Movement	0.16

Table 4. Average number of neighbors per communication slot for each node in our experiments over 100kmx100km field and radio range of 2km.

beacon-and-DR method is selected because it is the only method, besides per-node GPS, that can work in sparse networks. It requires an infrastructure where beacons provide known location to the nodes. Nodes, when in direct communication range of location beacons, update their location information based on the beacon's information. To make a fair comparison in the sparse situation, nodes without LOCALE also track their location with the DR system when not in range of the beacon. The primary difference is that LOCALE allows node-to-node position updates, while beacon-and-DR can only make updates when in direct contact with the beacon.

The base experiments are performed 100 times with 100 nodes. The results are sorted and the area errors at various percentiles are plotted. The tests are run on a 100 by 100 kilometers field. In most experiments, we adopted the random walk model, where speed and directions are reassessed every 8 minutes. In each 8 minute cycle, the nodes move in a random direction ranging a random distance uniformly distributed from 0 to 1 kilometer. At initialization, each node starts with the knowledge of its deployed location and their confidence levels are each set to a circle with unit radius.

For both LOCALE and the beacon-and-DR methods, we placed one fixed GPS beacon in the center of the field. The beacon has an accurate location, with a confidence set to a circle with 10 meters, and is not affected by erroneous information from other nodes. Movement tracking errors are taken from the results measured from our rudimentary tracking method described in Section 3.1. It incurs a maximum error of 20% in both X and Y directions. The radio range for the simulation is set to 2 kilometers. Only "in range" information is used to determine distance between two nodes during the transform phase. In every 8-minute cycle, a communication is attempted where LOCALE nodes search and merge with neighbors' location estimations.

The baseline, beacon-and-DR, method runs under the same conditions, with the only exception that nodes only receive location information from the location beacon. In the following sections, we demonstrate various aspects of LOCALE's performance.

4.1 Small Scale Example

In this section we start with a small scale simulation to provide intuition for LOCALE. 10 nodes were placed in a 10x10km area where each node was characterized by the parameters described before. Figures 6 and 7 show a

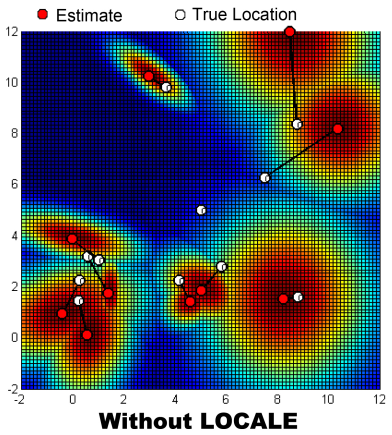

Figure 6. Top-down view of 10 nodes running the baseline beacon-and-DR method after 1000 cycles. Gradients represent confidence distributions, white dots show the estimated location and the connected dots show the actual locations. Some nodes exhibit large error because they have not been in range of the central location beacon for a long time.

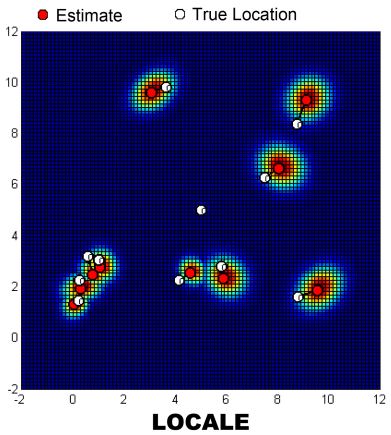

Figure 7. Top-down view of 10 nodes running LOCALE after 1000 cycles. Visual comparisons with Figure 6 show that LOCALE reduces both location error and uncertainty.

top-down view of the simulation field after 1000 simulated time slots, for the baseline beacon-and-DR method and LOCALE respectively. In the figures, the oval clouds show the estimated confidence for the location estimation. The white dots located in the center of the ovals are the position estimations themselves. The dots connected by a line segment to the position estimations are the true location of the node.

With the baseline method in Figure 6, the node's only source of information is the fixed beacon. Nodes that do not frequently encounter a beacon exhibit large errors. In contrast, Figure 7 has much smaller clouds, indicating that LOCALE's method offers much better (tighter) bounds on position estimates. This is because nodes that do not meet the beacon can still get location information from other nodes to refine their estimations. Furthermore, we see that with LOCALE, the uncertainty cloud more frequently encloses the true location. This indicates that, in LOCALE, the variance is a good estimate for the confidence of the location estimations. From these results, we see that even for relatively small areas, where the beacon has 13% coverage, LOCALE performs significantly better than the baseline beacon-and-DR method.

4.2 System Performance under Normal Conditions

In this section, we compare the performance of LOCALE with the baseline beacon-and-DR method for large-scale simulations. As described above, these simulations are performed 100 times, with 100 nodes over a 100x100 area, giving each node an average of 0.24 neighbors each simulated time period. The results of this experiment are shown in Figure 8 in term of median percent area errors described earlier. The error bars indicate the 25th percentile and the 75th percentile of the percent area error. Smaller values with tighter error bars are more preferable, since these indicate position estimates that are closer to the actual location.

Figure 8 shows that between 10,000 data points, the median area error for LOCALE is reduced by 21X to less than 0.2%. In addition, the 75th-percentile reduction improves further with 27X improvement to less than 0.5%. This graph shows that LOCALE drastically improves the usability of DR movement tracking with minimal beacon support, and allows for reasonably accurate localization without GPS.

4.3 Beacon's Influence on Accuracy

Our next experiments compare LOCALE to the baseline approach in terms of its sensitivity to a node's distance from a beacon. While the true coverage of the beacon is only the 2-km radio range, LOCALE's node-to-node information exchanges increase its effective coverage. These experiments are performed with the base parameters described before, but we now plot median error versus the node's distance from the beacon. Since the field is a 100x100 square with one beacon in its center, the maximum distance away from the beacon is around 70.

Figure 9 shows that both methods have lower uncertainty for nodes nearer to the beacon. The nodes running LOCALE, however, have a much lower error when compared to those with the beacon-and-DR method. At mid-range LOCALE effectively reduces the error by as much as 38X. The area error at the furthest distance from the beacon is less than 1%. This is because nodes that directly encounter the beacon can subsequently encounter other more remote nodes and propagate position information throughout the

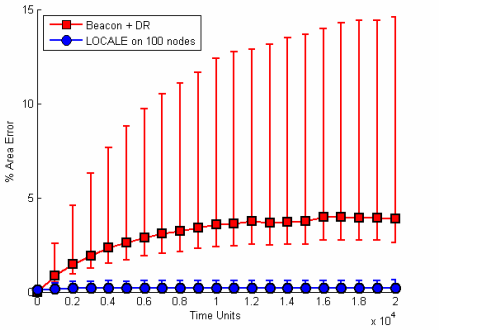

Figure 8. Plot of median area error under nodes running beacon-and-DR method and LOCALE. Error bars represent the 25th percentile and the 75th percentile. LOCALE's lower uncertainty values and tighter error bars indicate improved accuracy. In particular, there is up to 21X reduction in median uncertainty.

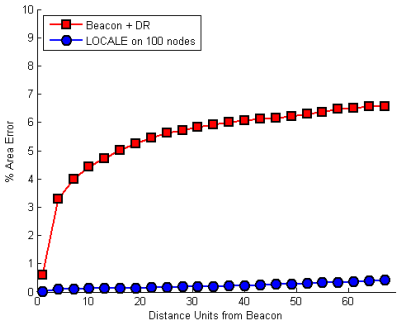

Figure 9. Plot of error with relation to distance away from the central beacon. Shows error reduction by 38X in the mid range.

field. This experiment shows that LOCALE significantly increases the effective coverage of the beacon and effectively spreads its information.

4.4 Error Correction with Unknown Initial Position

LOCALE's ability to propagate accurate position information node-by-node through the network effectively increases the coverage of the beacons. One question is how quickly this propagation can occur. That is, if all the nodes are deployed with unknown initial positions, how quickly can they converge towards reasonable accuracy and confidence? The simulations here use the same parameters as prior experiments, except that the initial position estimates are random within the 100x100 grid with the same initial confidence (a circle with unit radius) as the previous simulations.

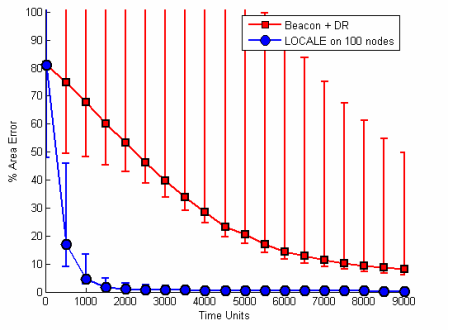

Figure 10. Plot of median area error under nodes running beacon method and LOCALE without initial position. Shows up to 57X median improvement after 2000 time slots.

Figure 10 shows the resulting median area error over time. We see that both methods show a decreasing error as more nodes encounter beacons over time. However, because of its node-to-node interactions, LOCALE propagates the beacon information throughout the network much more quickly. 75% of the nodes drop below 5% error within 2000 time intervals. At this point, the median percent area error is 57X lower compared with the beacon-and-DR method.

4.5 Effect of Node Density

We now explore the performance of LOCALE under varying node densities. While LOCALE is intended to improve localization resilience when networks are sparse, its accuracy improves as density increases. An ability to withstand highly varying densities is important for several reasons. Over a network's lifetime its density may vary greatly, due to phased deployments, varying node lifetimes, or simply movement patterns. The experiment was performed with the same parameters as the base case. All experiments were run on a 100x100 field and only the number of nodes on the field was varied between runs.

Figure 11 shows the median percent area error for the beacon-and-DR method, for a sparse configuration of LOCALE with only 10 nodes, and for a denser configuration of LOCALE with 100 nodes. Even with only 10 nodes, which have 0.02 neighbors per communication, LOCALE already shows a nearly 4X improvement relative to the baseline. 75% of LOCALE nodes performed better than the median of the baseline beacon-and-DR method. In the 100-node experiment, the density of nodes increases to 0.24 neighbors on average, and LOCALE has more neighbors to encounter. This increase in neighbor density directly results in even further improvements of position estimates. This experiment shows that LOCALE works for both dense and sparse situations. As a rule of thumb, for LOCALE to provide useful location estimation, density should be kept above 0.1 neighbors per communication. While its performance becomes better as node density increases, the 4X improvement is already significant in extremely sparse situations.

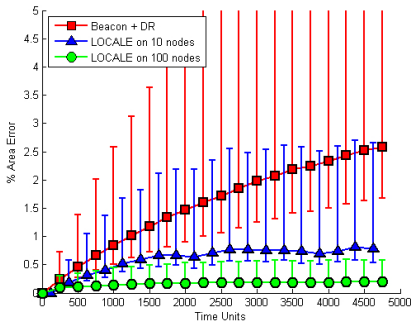

Figure 11. Plot of median error for different node densities. Shows error reduction even with a very sparse 10-node network.

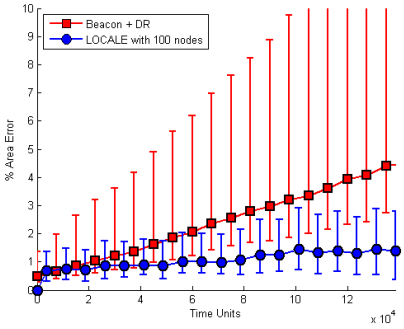

Figure 12. Plot of median error with nodes running LOCALE and the beacon-and-DR method respectively the extremely low-mobility ZebraNet movement model. Shows reduction of error even in this unfavorable case.

4.6 Effect of Highly Correlated Movement Model

LOCALE relies on node movement to propagate beacon information to other nodes with the assumption of identical-independent-distributions (IID) in movement models. This assumption of uncorrelated motion allows us to use Gaussian summation properties in our position confidence estimate. We next wish to explore whether LOCALE experiences greater error when the IID assumption does not fully hold. To explore this, we used the ZebraNet movement model [42], which is based on zebra movement traces obtained from ZebraNet deployment during the summer of 2005. The ZebraNet movement model is bi-modal, where nodes remain stationary for long periods of time, to feed, while, at other times, make long-range movements between feeding fields. In addition, the movement model is also correlated, where nodes tend to stay in groups with the same neighbors for long periods of time.

Figure 12 shows the results from experiments using this mobility model as the basis for how nodes move through the simulated grid. This, in turn, dictates how often nodes

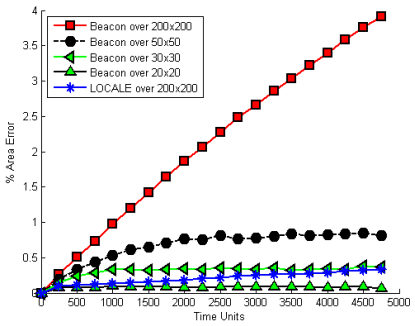

Figure 13. Plot of median area error of nodes in fields with different beacon densities. Shows LOCALE with one beacon in a 200x200 area obtains an accuracy of between that of the beacon method with a beacon every 20x20 and 30x30 area, a 64X infrastructure reduction.

encounter the fixed beacon, how often they encounter each other as neighbors, and how uncorrelated these contacts are. Although the sparseness and reduced mobility does adversely affect LOCALE's accuracy, it still gains more than 3X area error improvement over the beacon-and-DR method. At the end of simulation time, 75% of the nodes had position error of less than 25% of the nodes with beacon-and-DR. This is because the merge history table sets a merge delay of 100 time slots, for nodes that remain in range in 4 consecutive time periods. This experiment shows that even under difficult conditions that do not match the initial system assumptions—highly correlated movements, low mobility, and relatively fewer unique neighbor contacts—LOCALE is still able to greatly outperform the base case.

4.7 Beacon Density Requirements

In the deployment of sensor networks, cost of infrastructure is another general concern. In rural or remote areas where deploying a dense infrastructure would be impractical, it is desirable for a system to have as few requirements as possible. This experiment is designed to compare the accuracy of LOCALE and the beacon-and-DR method when different beacon densities are considered.

For LOCALE, the field is increased to 200x200km with one beacon in the center. For the beacon-and-DR method, experiments were run with beacons evenly distributed on the 200x200 field at different densities: one per 200x200, 50x50, 30x30, and 20x20. Figure 13 shows that, as beacon density goes up, the error for the beacon-and-DR method is reduced. More importantly, the same graph shows that nodes running LOCALE even with only 1 beacon over the entire 200x200 area can achieve better accuracy than the beacon-and-DR method with much costlier and higher beacon densities. In particular, in a system running LOCALE, beacons have a coverage area that is effectively 64X larger than the baseline approach. As a rule of thumb, beacon cov-

erage can be reduced to 0.1% for highly mobile situations, and even further reduced if placed in high-trafficked areas. By requiring fewer beacons, LOCALE systems can operate accurately with much lower infrastructure costs.

5 Related Work

There has been a wide range of research related to localization. These fields include localization methods in dense, sparse, and more capable robotics systems. In this section we discuss some of the related work in each of these categories and how they relate to and differ from LOCALE.

Localization for Fixed or Dense Mobile Networks: Much work has focused on fixed, dense sensor networks [19]. The nodes in such networks collaboratively measure relative distance and use this information to triangulate and calculate the network topology. Most methods use existing radio ranging for these measurements [1][2][23][25], while some only use connectivity as a metric [13][27][34]. Localization within a graphical model framework has also been explored [15]. In these methods, similar to LOCALE, nodes combine information from multiple potentially mobile sensor nodes in a dense network. While these methods are well suited for sensor nodes within a dense network, these localization methods incur a large communication overhead especially when nodes are mobile, as this topology information would need to propagate through the network. Furthermore, unlike LOCALE, these methods require full connectivity and cannot maintain location information in disconnected operations; such approaches would not function in our design space.

In systems where lower density localization is needed, some research has investigated the use of beacon nodes that send calibration signals for nodes without GPS to localize [24]. These methods are similar to the base beacon-and-DR method that we use to compare with LOCALE. While these methods remove the requirements for high density and specialized sensors, they incur the additional cost of a dedicated infrastructure. In other deployments, nodes cannot rely on inter-node measurements to obtain a position estimate due to lack of available connections. To solve this problem, some work uses mobile beacons where nodes can use to obtain their location information [29][31][36]. These are similar to LOCALE in the sense that they utilize movement to transfer location information, but LOCALE also further exploits position information from non-beacon nodes to propagate location information beyond the beacons.

Self-Localization in Mobile Robotics: Many approaches have been proposed to tackle the localization problem for autonomous robots. Work relevant to LOCALE can be largely categorized into two main categories: self-localization, and target localization. In self-localization, robots refine their location estimations based on sensor readings [3][4][11][28][38][40][41]. These techniques are similar to LOCALE in that they merge multiple observations of a fixed map or fixed targets to self localize. However, these techniques are based on multiple observations of a single node, as opposed to LOCALE where multiple nodes make multiple observations of each other's estimates and collaboratively form better self estimations. Furthermore, algorithm needed to recognize maps or objects are too computationally intensive, hence not directly applicable to sensor networks.

Robotic target-localization shares many of the same challenges as LOCALE in that they both use sensor readings to locate and track targets, and then rely on collaboration to merge observations in order to improve accuracy [12][35][39]. LOCALE differs, however, in that each nodes only observe themselves, as opposed to a commonly visible target. More importantly, LOCALE enables sparse collaboration, allowing nodes that cannot make direct observation of the localization target to collaboratively localize in a delay-tolerant manner.

6 Conclusion

In this paper, we introduced LOCALE, a distributed localization algorithm designed for sparse mobile sensor networks. We have shown that LOCALE greatly reduces the localization error, reducing the 75th-percentile area error by as much as 27X. Furthermore, when compared to the existing method relying on direct beacon contact and a dead-reckoning system, nodes using LOCALE achieved a similar location accuracy with 64X lower beacon density. LOCALE enables collaborative localization in sparse networks by not only allowing nodes to predict their current position but also to actively refine their location estimation from neighbor nodes.

LOCALE is a comprehensive system that, when compared to currently available methods, drastically improves localization capabilities of sparse mobile sensor networks. We have shown that it is a powerful, efficient and effective method that can be easily implemented in many types of hardware including low-capability sensor nodes. It provides a viable solution to the localization problem in mobile networks. More broadly, LOCALE effectively enables and utilizes collaboration localization in sparse and disconnected mobile sensor networks.

References

[1] P. Bergamo and G. Mazzimi. Localization in Sensor Networks with Fading and Mobility. In *Proceedings of IEEE PIMRC*, 2002.

[2] N. Bulusu, J. Heidemann, and D. Estrin. Gps-less Low Cost Outdoor Localization for Very Small Devices. *IEEE Personal Communications Magazine*, 7(5):28–34, October 2000.

[3] W. Burgard, A. Derr, D. Fox, and A. Cremers. Integrating Global Position Estimation and Position Tracking for Mobile Robots: the Dynamic Markov Localization approach. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '98)*, pages 730–735, October 1998.

[4] W. Burgard, D. Fox, D. Hennig, and T. Schmidt. Estimating the Absolute Position of a Mobile Robot Using Position ProbabilityGrids. In *Proc. of the National Conferenceon Artificial Intelligence*, 1996.

[5] Crossbow. MTS-MDA Series Users Manual. http://www.xbow.com/, June 2006.
[6] Crossbow. Stargate DataSheet. http://www.xbow.com/, Nov. 2006.
[7] Crossbow. IMote2 DataSheet. http://www.xbow.com/, 2007.
[8] Crossbow. MICA2 DataSheet. http://www.xbow.com/, 2007.
[9] Crossbow. MICAZ DataSheet. http://www.xbow.com/, 2007.
[10] Crossbow. TelosB DataSheet. http://www.xbow.com/, 2007.
[11] A. Curran and K. J. Kyriakopoulos. Sensor-based Self-localization for Wheeled Mobile Robots. In *IEEE International Conference on Robotics and Automation*, May 1993.
[12] M. Dietl, J. Gutmann, and B. Nebel. Cooperative Sensing in Dynamic Environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'01)*, 2001.
[13] T. He, C. Huang, B. Blum, J. Stankovic, and T. Abdelzaher. Range-Free Localization Schemes in Large Scale Sensor Networks. In *MobiCom '03*, 2003.
[14] B. Hull, V. Bychkovsky, Y. Zhang, K. Chen, M. Goraczko, A. K. Miu, E. Shih, H. Balakrishnan, and S. Madden. CarTel: A Distributed Mobile Sensor Computing System. In *4th ACM SenSys*, November 2006.
[15] A. T. Ihler, J. W. Fisher III, R. L. Moses, and A. S. Willsky. Nonparametric belief propagation for self-calibration in sensor networks. In *Information Processing in Sensor Networks*, 2004.
[16] P. Juang, H. Oki, Y. Wang, et al. Energy-Efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with ZebraNet. In *Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-X)*, Oct. 2002.
[17] Y.-B. Ko and N. H. Vaidya. Location-Aided Routing (LAR) in Mobile Ad Hoc Networks. In *Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom98)*, 2005.
[18] B. Kusy, G. Balogh, A. Ledeczi, and M. M. J. Sallai. inTrack: High Precision Tracking of Mobile Sensor Nodes. In *4th European Workshop on Wireless Sensor Networks (EWSN 2007)*, January 2007.
[19] K. Langendoen and N. Reijers. Distributed Localization in Wireless Sensor Networks: a Quantitative Comparison. *Computer Networks: The International Journal of Computer and Telecommunications Networking*, 43(4):499–518, 2003.
[20] W. E. Mantzel, C. Hyeokho, and R. G. Baraniuk. Distributed Camera Network Localization. In *Signals, Systems and Computers*, 2004.
[21] M. Mauve, J. Widmer, and H. Hartenstein. A Survey on Position-Based Routing in Mobile Ad-Hoc Networks. *IEEE Network Magazine 15 (6), pp. 30-39*, Nov. 2001.
[22] Maxstream, Inc. XTend OEM RF Module: Product Manual v1.2.4. http://www.maxstream.net/, Oct. 2005.
[23] D. Moore, J. Leonard, D. Rus, and S. Teller. Robust Distributed Network Localization with Noisy Range Measurements. In *Proc. 2nd ACM SenSys*, pages 50–61, Baltimore, MD, November 2004.
[24] R. Moses, D. Krishnamurthy, and R. Patterson. A Self-Localization Method for Wireless Sensor Networks. In *Eurasip Journal on Applied Signal Processing, Special Issue on Sensor Networks*, 2002.
[25] R. Nagpal, H. Shrobe, and J. Bachrach. Organizing a Global Coordinate System from Local Information on an Ad Hoc Sensor Network. In *2nd International Workshop on Information Processing in Sensor Networks (IPSN '03)*, Apr. 2003.
[26] D. Niculescu and B. Nath. Ad Hoc Positioning System (APS) using AoA. In *In Proceedings of INFOCOM 2003*, 2003.
[27] D. Niculescu and B. Nath. DV Based Positioning in Ad hoc Networks. In *Journal of Telecommunication Systems*, 2003.
[28] C. Olson. Probabilistic self-localization for mobile robots. In *IEEE Transactions on Robotics and Automation, vol. 16, no. 1, pp. 55-66, Feb. 2000.*
[29] P. N. Pathirana, N. Bulusu, S. Jha, and A. V. Savkin. Node Localization Using Mobile Robots in Delay-Tolerant Sensor Networks. In *IEEE Transactions on Mobile Computing*, volume 4, pages 285–296, May 2005.
[30] PNI Corporation. MicroMag3 DataSheet. https://www.pnicorp.com/, 2007.
[31] N. B. Priyantha, H. Balakrishnan, E. Demaine, and S. Teller. Mobile-Assisted Localization in Wireless Sensor Networks. In *IEEE INFOCOM*, Miami, FL, March 2005.
[32] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The Cricket Location-Support system. In *6th ACM MOBICOM*, Aug. 2000.
[33] Servoflo Corporation. AMI601 DataSheet. http://www.servoflo.com/, 2007.
[34] Y. Shang, W. Ruml, Y. Zhang, and M. Fromherz. Localization From Mere Connectivity. In *MobiHoc'03*, 2003.
[35] R. C. Smith and P. Cheeseman. On the Representation and Estimation of Spatial Uncertainty. *The International Journal of Robotics Research*, 5(4):56–68, 1986.
[36] K.-F. Ssu, C.-H. Ou, and H. C. Jiau. Localization with Mobile Anchor Points in Wireless Sensor Networks. In *Vehicular Technology, IEEE Transactions on*, 2005.
[37] STMicroelectronics. LIS3L02AQ DataSheet. http://www.st.com/, Nov. 2004.
[38] A. Stroupe and T. Balch. Collaborative Probabilistic Constraint-Based Landmark Localization. In *Proceedings of IROS '02*, October 2002.
[39] A. Stroupe, M. Matrin, and T. Balch. Distributed Sensor Fusion for Object Position Estimation by Multi-robot Systems. In *Int. Conf. on Robotics and Automation (ICRA'01)*, 2001.
[40] S. Thrun. Bayesian Landmark Learning for Mobile Robot Localization. In *Machine Learning*, volume 33, pages 41–76, 1998.
[41] S. Thrun, D. Fox, W. Burgard, and F. Dellaert. Robust Monte Carlo Localization for Mobile Robots. *Artificial Intelligence*, 128(1-2):99–141, 2000.
[42] Y. Wang, P. Zhang, T. Liu, C. Sadler, and M. Martonosi. Movement Data Traces from Princeton ZebraNet Deployments, 2007. CRAWDAD Database. http://crawdad.cs.dartmouth.edu/.
[43] G. Welch and E. Foxlin. Motion Tracking: No Silver Bullet, but a Respectable Arsenal. *IEEE Computer Graphics and Applications, special issue on Tracking*, 22(6):24–38, Nov. 2002.
[44] Xemics. DP1201A, 433.92MHz Drop-in Module Product Brief. http://www.xemics.com/, Mar. 2004.
[45] P. Zhang, C. Sadler, S. Lyon, and M. Martonosi. Hardware Design Experiences in ZebraNet. In *Proceedings of the ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2004.