Exploiting Temporal Stability and Low-Rank Structure for Localization in Mobile Networks

Swati Rallapalli Lili Qiu Yin Zhang Yi-Chao Chen
The University of Texas at Austin
{swati,lili,yzhang,yichao}@cs.utexas.edu

ABSTRACT

Localization is a fundamental operation for many wireless networks. While GPS is widely used for location determination, it is unavailable in many environments either due to its high cost or the lack of line of sight to the satellites (*e.g.*, indoors, under the ground, or in a downtown canyon). The limitations of GPS have motivated researchers to develop many localization schemes to infer locations based on measured wireless signals. However, most of these existing schemes focus on localization in static wireless networks. As many wireless networks are mobile (*e.g.*, mobile sensor networks, disaster recovery networks, and vehicular networks), we focus on localization in mobile networks in this paper. We analyze real mobility traces and find that they exhibit temporal stability and low-rank structure. Motivated by this observation, we develop three novel localization schemes to accurately determine locations in mobile networks: (i) Low Rank based Localization (LRL), which exploits the low-rank structure in mobility, (ii) Temporal Stability based Localization (TSL), which leverages the temporal stability, and (iii) Temporal Stability and Low Rank based Localization (TSLRL), which incorporates both the temporal stability and the low-rank structure. These localization schemes are general and can leverage either mere connectivity (*i.e.*, range-free localization) or distance estimation between neighbors (*i.e.*, range-based localization). Using extensive simulations and testbed experiments, we show that our new schemes significantly outperform state-of-the-art localization schemes under a wide range of scenarios and are robust to measurement errors.

Categories and Subject Descriptors

C.2.1 [**Computer Communication Networks**]: Network Architecture and Design—*Wireless communication*

General Terms

Algorithms, Experimentation, Measurement

Keywords

Localization, Mobility, Temporal Stability, Low-rank Structure

1. INTRODUCTION

Motivation: Knowledge of location is critical to many wireless network applications, such as geographic routing [25, 41], context-aware applications [26, 27], environment surveillance [1, 49, 52], habitat monitoring [11, 24], vehicle tracking [48], disaster recovery [14], military reconnaissance [20] and underwater surveillance [4, 51]. Meanwhile, many wireless networks are mobile. For example, wireless devices move with rescuers in a disaster area, move with soldiers in a battlefield, move with tourists in a visitor center, move with animals in a habitat, move with water in the ocean, and move with vehicles around a city. Even sensor networks that used to be static are becoming mobile in order to leverage mobility to efficiently cover a large area using a small number of sensors (*e.g.*, [34, 35, 38]). It is therefore important to develop effective methods to accurately determine the locations of nodes in mobile networks.

Existing approaches and limitations: The Global Positioning System (GPS) [18] is widely used to obtain location information in wireless networks. But GPS does not work in many environments due to the lack of line of sight to the satellites (*e.g.*, indoors, under the ground, or in a downtown canyon). Moreover, it is often too expensive to equip every wireless device with a GPS.

The limitations of GPS have motivated researchers to develop many localization schemes to infer locations based on cheap hardware and wireless measurements (*e.g.*, [3, 21, 22, 29, 37, 46, 47, 55]). For these schemes, localization accuracy depends heavily on the amount of information that one can extract to constrain the set of possible locations that a given node belongs to.

Most existing localization schemes are designed for static wireless networks. For example, there are many interesting proposals on deriving location constraints from measurements of wireless network connectivity [46, 47], signal strength [3], angle of arrival [37], and difference in arrival time of different types of signals [40]. To localize nodes in mobile networks, one possible approach is to treat a mobile network during different time intervals as separate static networks and apply existing localization schemes for static networks to the mobile network at each interval. However, such an approach is unlikely to achieve high accuracy because (i) mobility-induced fading can make distance measurements much less accurate in mobile networks, and (ii) it is difficult to conduct synchronized measurements when nodes are moving.

There are only a limited number of recent works that explicitly consider the problem of localization in mobile networks (*e.g.*, [2, 22, 42, 50]). However, these works do not fully exploit the location information available in each time interval. For instance, existing localization schemes for mobile networks only use the maximum speed to constrain the distance between a node's positions during two consecutive intervals. Moreover, their evaluation uses simulation based on synthetic mobility traces. It is not clear how well they perform under a real mobility pattern or in a real network.

Our approach: In this paper, we develop novel techniques to accurately localize nodes in mobile networks by exploiting structural properties of the underlying node mobility patterns. Intuitively, the nodes' coordinates over time are not independent but have certain relationships. By exploiting such relationships, we can derive additional constraints to further narrow down the set of feasible locations and thus significantly improve localization accuracy.

To extract the relationships between nodes' coordinates across different time intervals, we first analyze the characteristics of a number of real mobility traces, including (i) taxi and bus traces in San Francisco [9], Shanghai [62], and Seattle [45], (ii) human mobility traces [23], and (iii) ZebraNet traces [60]. Interestingly, while these traces capture rather different mobility scenarios, they all exhibit two pronounced structural properties: (1) *low-rank structure*, *i.e.*, the matrix formed by node coordinates over time can be well approximated by a low-rank matrix; and (2) *temporal stability*, *i.e.*, the direction and speed of the same mobile node is often similar at adjacent intervals. Our results suggest that these structural properties are also present in synthetic traces generated using two commonly used mobility models: the standard random waypoint model [7] and the modified random waypoint model [22].

Motivated by these observations, we develop a general framework that simultaneously incorporates location constraints during each interval and exploits the low-rank structure and temporal stability in mobility. Specifically, we formulate the localization problem as an optimization problem that minimizes (i) the fitting error between measured distance and estimated distance based on node coordinates, (ii) the error in approximating the estimated coordinate matrix using a low-rank matrix, and (iii) the total changes in nodes' velocity during adjacent time intervals. Essentially, (i) leverages the location information during each time interval of the mobile network, which can be obtained using existing localization techniques for static networks (*e.g.*, [3, 37, 40, 47]), (ii) reflects the low-rank nature of the coordinate matrix in a mobile network, and (iii) captures the temporal stability in node movement.

Using this framework, we develop three novel localization schemes for mobile networks: (i) Low Rank based Localization (LRL), which exploits the low-rank structure in mobility, (ii) Temporal Stability based Localization (TSL), which leverages the temporal stability, and (iii) Temporal Stability and Low Rank based Localization (TSLRL), which simultaneously takes into account the temporal stability and the low-rank structure of the mobility. These localization schemes use distance estimation between neighbors and we call them *range-based* localization schemes. In addition, based on the general framework, we also develop *range-free* variants of these schemes, namely, LRL-RF, TSL-RF, and TSLRL-RF. These variants use only network connectivity for localization and can therefore support nodes that are unable to obtain accurate distance estimation (*e.g.*, due to lack of RSS measurements).

We extensively evaluate our localization schemes using (i) simulations based on both synthetic mobility traces and several real mobility traces, and (ii) experiments in a sensor network testbed consisting of 25 or 36 $mica2$ motes. The simulation results show that our new schemes significantly out-perform the existing schemes over a wide range of scenarios. For example, TSLRL reduces the mean absolute error by a factor of 2.3-63.5 over Centroid (*i.e.*, the mean absolute error of Centroid is 2.3–63.5 times of that in TSLRL), 1.1-20.4 over MDS, 1.6-27.2 over Sextant, 1.8-49.5 over MCL, and 1.2-17.3 over MSL* with and without measurement noise, where Centroid, MDS, and Sextant are well-known localization schemes for static networks, MCL and MSL* are state-of-the-art localization schemes for mobile networks. TSL sees similar performance improvement. We further compare various localization schemes using testbed experiments, and find that our schemes reduce the mean absolute error by a factor of 2.5-4.1 over Centroid, 1.6-2.3 over MDS, 2.4-3.9 over Sextant, 2.8-4.4 over MCL, and 2.1-3.2 over MSL*. Moreover, our schemes are robust and can achieve high localization accuracy even in the presence of measurement noise, irregular topologies, and different mobility patterns.

Paper organization: The remainder of the paper is organized as follows. In Section 2, we review the related work. We analyze the characteristics of real mobility traces in Section 3. We formulate the problem of localization in mobile networks in Section 4, and describe our solution in Section 5. In Section 6 and Section 7, we evaluate our approach using both simulation and testbed. We conclude in Section 8.

2. RELATED WORK

We classify the related work into the following three categories: (i) localization in static wireless networks, (ii) localization in mobile networks, and (iii) general compressive sensing.

2.1 Localization in Static Wireless Networks

Most works on localization target static wireless networks. The first class of works in this category focus on localization in static single-hop wireless networks. For example, [8] proposes Centroid, which estimates the location of a node as the center of all neighboring anchor nodes. When there are no anchor nodes nearby, a node estimates its location as the center of the area. RADAR [3] uses signal strength measurements from multiple base stations to locate and track users. Cricket [40] obtains accurate distance estimates based on the difference between the arrival time of radio and ultrasound signals. VORBA [37] leverages angle of arrival measurements from 802.11 base stations. [29] exploits the signal strength information from a group of clients to simultaneously locate all of them instead of one client at a time. See [21] for an extensive survey on this subject.

The second class focuses on localization in static multi-hop wireless networks. For example, Savvides *et al.* [43] develop a distributed localization approach that iterates through two phases: ranging and estimation. During the ranging phase, each node estimates its distance to its neighbors; and during the estimation phase, it estimates its location based on the ranging information and the locations of its neighbors whose positions have been determined. To limit error accumulation in [43], Savvides *et al.* [44] formulate the localization problem as a global non-linear optimization problem. Shang *et al.* [47] apply multi-dimensional scaling (MDS) to determine location in a centralized fashion. MDS estimates the distance matrix by computing the shortest path distance between each pair of nodes and then finds the nodes' coordinates by solving an optimization problem: $\min_{x_1,\ldots,x_n} \sum_{i<j}(\|x_i - x_j\| - d_{ij})^2$, where $\|x_i - x_j\|$ is the Euclidean distance between two locations x_i and x_j, and d_{ij} is the measured distance. MDS performs poorly when the shortest path distance does not correlate well with the Euclidean distance, which is common in irregularly shaped networks. In [36], the authors propose robust quadrilateral for localization. It finds sets of four nodes that are fully connected, and localizes the fourth node based on the positions of the other three nodes. Robust quadrilateral conditions have to be satisfied by the fourth node in order to prevent error accumulation. Therefore it improves accuracy at the cost of leaving some nodes un-localized. Biswas *et al.* [6] formulate the localization problem as a semidefinite program, and later develop global optimization approaches [5]. Wang *et al.* [54] further propose a low-rank semi-definite programming formulation of the localization problem. Inspired by these localization schemes for static networks, we focus on localization in mobile networks.

Unlike most of the previous approaches, which represent inferred locations using points, Sextant [19] denotes inferred locations as re-

Trace name	Description	Matrix size	Time interval
Cabspotting traces [9]	Taxis in San Francisco Bay Area over a month	100 nodes $\times$ 300 intervals	1 minute
Shanghai taxi traces [62]	Taxis in Shanghai on 1 day	100 nodes $\times$ 300 intervals	1 minute
Seattle bus traces [45]	Buses in Seattle during 2001	545 nodes $\times$ 300 intervals	1 minute
ZebraNet traces [60]	ZebraNet deployment in 2005 summer	61 nodes $\times$ 90 intervals	8 minutes
Human mobility traces [23]	several students' movement in KAIST	92 nodes $\times$ 499 intervals	30 seconds

Table 1: Real traces used in mobility characterization.

gions that satisfy distance measurements. In particular, when node i hears from node j, it extracts a positive constraint that their distance is within the communication range. When node i does not hear from node j, it extracts a negative constraint that their distance is larger than the communication range. A node then estimates its location by finding a region that satisfies both positive and negative constraints. All points in a region are considered equally likely locations for a node. [53] proposes probabilistic region-based localization that uses a dynamic mesh to represent a region and computes the probability for a node to reside in different parts of the region. This improves accuracy over Sextant at the cost of higher computation time.

2.2 Localization in Mobile Networks

Compared to significant related work on static network localization, there are considerably fewer works on localization in mobile networks. Among the few existing works, [22] is the first localization scheme for mobile networks. It uses a sequential Monte Carlo Localization (MCL) method to localize mobile sensors. A node uses its previous location and maximum speed to generate possible current coordinates. Then it filters out infeasible locations using the current connectivity information. Since MCL only extracts information for nodes that are either direct neighbors or 2-hop neighbors from anchor nodes, it requires a high anchor density to work well. Moreover, its sampling technique converges very slowly as reported in [42] and observed in our own evaluation. [2, 42, 50] propose several enhancements over MCL sampling. Among them, MSL* [42] performs the best. MCL only supports mobile networks, and MSL* [42] modifies the sampling procedure to support both static and mobile networks. Moreover, it lets nodes use information only from the neighbors that have more accurate coordinates to speed up convergence and improve accuracy.

2.3 Compressive Sensing

The localization problem is related to the general area of compressive sensing in that both aim to recover the unknowns based on partial observations. To cope with the under-determined nature of the problems, many algorithms have been proposed in the area of compressive sensing. Their effectiveness depends on their ability to exploit the unique structure in the data. For example, [15, 16, 39] exploit sparsity, [10, 17] exploit low rank structure, and [61] exploits spatio-temporal properties. Motivated by these works, our work exploits the low rank and temporal stability of coordinate matrices for localization in mobile networks for the first time.

3. MOBILITY CHARACTERIZATION

In this section, we analyze the characteristics of mobility patterns using both real and synthetic traces, and find that they often exhibit low-rank structure and temporal stability.

Low-rank structure: Our first finding concerns the low-rank nature of the coordinate matrix. Specifically, consider n wireless nodes in a 2-dimensional Euclidean space. Let M be the $2n \times t_{\max}$ coordinate matrix over time, where $M(i,t)$ and $M(i+n,t)$ denote node i's x-coordinate and y-coordinate at time t ($1 \leq t \leq t_{\max}$), respectively. It is not difficult to see that if every node moves at a constant velocity, even when different nodes move at different velocities, the coordinate matrix M always has rank 2. This is because we always have $M(:,t) = \mathbf{z} + t \cdot \mathbf{v}$, where $\mathbf{z}$ is a column vector that represents the initial coordinates of all the nodes, and $\mathbf{v}$ is a column vector that gives the velocities of all the nodes, and t is the current time. As a result, M can be represented as the sum of two rank-1 matrices: $M = \mathbf{z} \cdot \mathbf{1}^T + \mathbf{v} \cdot \mathbf{t}^T$, where $\mathbf{1}$ is an all-1 column vector, and $\mathbf{t} = [1, 2, \cdots, t_{\max}]^T$ is the time vector.

In practice, a node may not always move at a constant velocity. However, it is common that a node may travel at a constant velocity for some time before it changes direction or speed. This suggests that it is quite likely that the real coordinate matrix M exhibits low-rank structure. To validate this conjecture, we analyze a number of publicly available traces as shown in Table 1. Here the time interval is determined by the frequency of trace collection (*e.g.*, the vehicular traces recorded GPS readings of the vehicles around once a minute). Since the human mobility traces and ZebraNet traces have only a few nodes, we pre-process these traces as follows. The KAIST campus traces were taken from 4 students living in a campus dormitory. There are 92 trace files, each of which represents a daily trace from one participant. We treat each file as a trace from a distinct person, which gives us 92 nodes. We use ZebraNet traces from the second deployment, which has data from 5 zebras. We generate 61 synthetic zebras from these 5 real zebras by partitioning each trace over time and treating each partition as a synthetic zebra. This gives us 61 zebras over 90 time intervals. Since the original vehicular traces already contain a large number of nodes over an extended period, such pre-processing is not needed. Also note that some of these mobility traces contain a small fraction of missing values. We fill in these missing values using nearest-neighbor interpolation [56].

For comparison, we also generate synthetic mobility traces for 50 nodes in a $200m \times 200m$ area over 100 time intervals using the standard random waypoint model [7] and the modified random waypoint model proposed in [22]. The random waypoint model is one of the most widely used mobility models. In the standard random waypoint model, each node picks a random location as a destination and moves towards the destination at a randomly selected velocity; after reaching the destination, the node pauses for some random amount of time and selects a new destination and repeats the process. [22] proposes a modified random waypoint to overcome the limitation of the standard random waypoint model, which experiences decay in average speed, as reported in [59]. To maintain the average speed, in the modified random waypoint model, each node randomly chooses a speed every interval (instead of staying at the same speed until reaching the destination as in the standard random waypoint model). We use a maximum velocity of 10 $m/interval$ for low mobility and 30 $m/interval$ for high mobility under both the standard and modified random waypoint models.

For each mobility trace, we first derive the corresponding coordinate matrix and mean center each row (*i.e.*, subtract from each row its mean value). We then apply singular value decomposition (SVD) to examine if the mean-centered coordinate matrix has a good low-rank approximation. The metric we use is the fraction of total variance captured by the top K singular values, *i.e.*, $\left(\sum_i^N s_i^2\right) / \left(\sum_i s_i^2\right)$, where s_i is the i-th largest singular value and $\left(\sum_i s_i^2\right)$ gives the total variance of the mean-centered coordinate

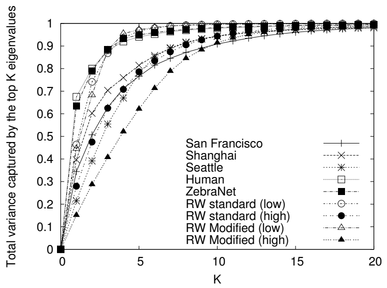


Figure 1: Low-rank structure in mobility traces.

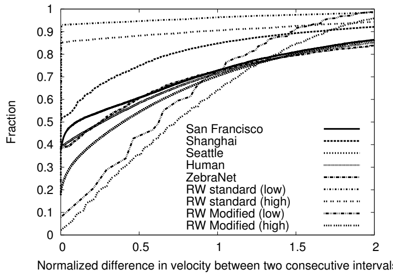


Figure 2: Temporal stability in mobility traces.

matrix. Note that $1 - \left(\sum_i^N s_i^2\right) / \left(\sum_i s_i^2\right)$ is the relative approximation error of the best rank-K approximation with respect to the squared Frobenius norm. Figure 1 plots the fraction of total variance captured by the top K singular values for different mobility traces. As we can see, in all traces most variance is captured by the top few singular values. For example, the top 5 singular values capture 76.6%-94.9% variance in the real traces and 62.1%-97.2% variance in the synthetic traces. These results clearly suggest that real mobility exhibits low-rank structure.

Temporal stability: We further analyze the temporal stability in these traces. For every node i and time interval t, we compute the normalized velocity change:

$$NVC(i,t) \triangleq \frac{\|\vec{v}(i,t) - \vec{v}(i,t-1)\|_2}{\text{mean}_{i,t}(\|\vec{v}(i,t)\|_2)},$$

where $\vec{v}(i,t)$ is node i's velocity vector (in the 2-dimensional Euclidean space) at time interval t, and $\|\cdot\|_2$ is the ℓ_2-norm (with $\|\vec{z}\|_2 = \sqrt{\sum_k \vec{z}(k)^2}$ for any vector $\vec{z}$).

Figure 2 plots the CDF of $\{NVC(i,t)\}$. As it shows, in four out of the five real traces, over 36% of time the velocity remains the same for two consecutive intervals (*i.e.*, $NVC = 0$), and over 42% of time NVC is within 10%. The only exception is Seattle bus traces, which have 34% of time with $NVC \le 10\%$. This is likely due to frequent bus stops. In the standard random waypoint traces, $NVC \le 10\%$ for 86%-93% of time. The modified random waypoint traces experience lower stability because every node chooses a random velocity at every time interval. These results indicate that the real mobility often exhibits temporal stability, *i.e.*, nodes move at a constant velocity for some time. The extent of temporal stability varies with the network environment.

Implication: The presence of low-rank structure and temporal stability in a wide range of mobility traces motivates us to explicitly exploit these structural properties to achieve better localization accuracy in mobile networks. Since the extent of low-rank structure and temporal stability may vary with the network environment, we cannot strictly enforce these properties in the localization solution. Instead we incorporate them into the objective function. Specifically, static localization schemes aim to minimize the fitting error between observed and estimated distance based on nodes' coordinates. Now we further minimize (i) the approximation error between the real coordinate matrix and its low-rank approximation, and (ii) the total change in velocity during consecutive intervals.

4. PROBLEM FORMULATION

In this section, we mathematically formulate the problem of mobile sensor localization. Specifically, we consider N mobile sensors moving in a d-dimensional Euclidean space. These mobile sensors have no built-in GPS and thus need to be localized by our localization algorithm. Meanwhile, we assume that there are N_{anch} mobile anchor nodes that are equipped with built-in GPS and thus have known locations. We divide time into $t_{\max}$ equal-sized time intervals. During each time interval, we measure (i) the locations for the anchor nodes, (ii) the distance between the sensors, and (iii) the distance between the sensors and anchors. We then try to localize the mobile sensors based on such measurements.

Notations: Our formulation uses the following notations:

- $X(:,i,t)$ denotes the d-dimensional Euclidean coordinate for sensor i ($1 \le i \le N$) at time interval t ($1 \le t \le t_{\max}$).
- $A(:,a,t)$ denotes the d-dimensional Euclidean coordinate for anchor a ($1 \le a \le N_{\text{anch}}$) at time interval t ($1 \le t \le t_{\max}$).
- $D_{ij}(t) = \|X(:,i,t) - X(:,j,t)\|_2^2 = \sum_{k=1}^{d}(X(k,i,t) - X(k,j,t))^2$ is the *squared Euclidean distance* between sensor i and sensor j at time interval t.
- $C_{ia}(t) = \|X(:,i,t) - A(:,a,t)\|_2^2 = \sum_{k=1}^{d}(X(k,i,t) - A(k,a,t))^2$ is the *squared Euclidean distance* between sensor i and anchor a at time interval t.
- $D_{ij}^{\text{eq}}(t)$, $D_{ij}^{\text{ub}}(t)$, and $D_{ij}^{\text{lb}}(t)$ are the equality, upper bound, and lower bound constraints on $D_{ij}(t)$, respectively.
- $C_{ia}^{\text{eq}}(t)$, $C_{ia}^{\text{ub}}(t)$, and $C_{ia}^{\text{lb}}(t)$ are the equality, upper bound, and lower bound constraints on $C_{ia}(t)$, respectively.

Note that X is a 3-dimensional array. To formally capture the low-rank structure and temporal stability that we observe in Section 3, it is more convenient to represent X in the form of a 2-dimensional coordinate matrix, which can be obtained by collapsing the first two dimensions of X into a single dimension. Let $M = \mathsf{matricize}(X)$ be the resulted coordinate matrix. We have:

$$M(k + (i-1) * N, t) = X(k,i,t). \tag{1}$$

Incorporating distance measurements: We first formulate the localization problem for a static network by incorporating distance equality and bound constraints. We capture the total violation of distance constraints at time interval t using $f(X,t)$ shown below:

$$\begin{aligned} f(X,t) \triangleq\ & \textstyle\sum_{ij}\left(D_{ij}(t) - D_{ij}^{\text{eq}}(t)\right)^2 + \\ & \textstyle\sum_{ij}\min\left\{0, D_{ij}(t) - D_{ij}^{\text{lb}}(t)\right\}^2 + \\ & \textstyle\sum_{ij}\max\left\{0, D_{ij}(t) - D_{ij}^{\text{ub}}(t)\right\}^2 + \\ & \textstyle\sum_{ia}\left(C_{ia}(t) - C_{ia}^{\text{eq}}(t)\right)^2 + \\ & \textstyle\sum_{ia}\min\left\{0, C_{ia}(t) - C_{ia}^{\text{lb}}(t)\right\}^2 + \\ & \textstyle\sum_{ia}\max\left\{0, C_{ia}(t) - C_{ia}^{\text{ub}}(t)\right\}^2 \end{aligned} \tag{2}$$

Here the first and fourth terms quantify the total violation of the equality constraints: $D_{ij}(t) = D_{ij}^{\text{eq}}(t)$ and $C_{ia}(t) = C_{ia}^{\text{eq}}(t)$. The second and fifth terms capture the total violation of the lower bound constraints: $D_{ij}(t) \ge D_{ij}^{\text{lb}}(t)$ and $C_{ia}(t) \ge C_{ia}^{\text{lb}}(t)$. Similarly, the third and last terms represent the total violation of the upper bound constraints: $D_{ij}(t) \le D_{ij}^{\text{ub}}(t)$ and $C_{ia}(t) \le C_{ia}^{\text{ub}}(t)$.

A few comments follow. First, we can use RSS measurements to derive equality constraints, because RSS is a function of the distance between the sender and the receiver. Moreover, when two nodes can directly hear each other, we can use the communication range as an upper bound of the distance between them. If two nodes cannot hear each other, then the communication range becomes a lower bound of the distance between them. In addition, if two nodes cannot directly hear each other but are connected through some indirect path, we can apply the triangular inequality and use the shortest path distance (in terms of the estimated distance based on RSS) as the upper bound. Note that all the equality and bound constraints on the Euclidean distance need to be squared, as $D_{ij}(t)$ and $C_{ia}(t)$ are defined with respect to the *squared* Euclidean distance.

Second, to support range-free localization, which is useful when the nodes can only get connectivity information but not the exact distance estimation, we simply remove equality constraints from $f(X, t)$ and only retain the lower bound and upper bound constraints. The lower bound constraints remain the same as above. As for the upper bound constraints, since the shortest path distance in terms of the estimated distance based on RSS is unavailable, we use a looser upper bound of $H \times R$, where H is the shortest hop count and R is the communication range.

Third, due to measurement errors, both the equality constraints and the upper/lower bound constraints may not be strictly satisfiable. By using the total violation against these constraints as our optimization objective (instead of trying to strictly enforcing these constraints), we can always find feasible solutions.

Fourth, it is easy to see that function $\max\{0, z\}^2$ is continuously differentiable with respect to variable z and the gradient is $2 \cdot \max\{0, z\}$. Similarly, function $\min\{0, z\}^2$ is continuously differentiable with respect to variable z and the gradient is $2 \cdot \min\{0, z\}$. Therefore, $f(X, t)$ is a continuously differentiable function of variables $X(:, i, t)$. This allows us to apply a gradient-based algorithm to minimize $f(X, t)$ (see Section 5).

Incorporating temporal stability constraints: To capture the temporal stability of sensor mobility, we introduce a temporal transformation matrix T and define a penalty function as follows:

$$g(X) \triangleq \left\| M * T^T \right\|_F^2, \tag{3}$$

where $\|\cdot\|_F$ is the Frobenius norm (with $\|Z\|_F = \sqrt{\sum_{ij} Z(i,j)^2}$ for any matrix Z), and $M = \mathsf{matricize}(X)$ is the coordinate matrix obtained by collapsing the first two dimensions of X into a single dimension according to Eq. (1).

A simple choice of the temporal transformation matrix is $T = Toeplitz(0, 1, -2, 1)$, which denotes the Toeplitz matrix with central diagonal given by ones, the first upper diagonal given by negative twos, and the second upper diagonal given by ones, *i.e.*,

$$T = \begin{bmatrix} 1 & -2 & 1 & 0 & \cdots \\ 0 & 1 & -2 & 1 & \ddots \\ 0 & 0 & 1 & -2 & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots \end{bmatrix}. \tag{4}$$

This temporal transformation matrix intuitively expresses the fact that the direction and the speed of the same mobile sensor are often similar at adjacent points in time. As a result, one can well approximate a sensor's location at time interval t using the mid-point of its locations at time intervals $t-1$ and $t+1$. With this T, we have:

$$g(X) = \sum_{k,i,t} \left(X(k,i,t-1) + X(k,i,t+1) - 2 \cdot X(k,i,t) \right)^2,$$

where each term $\left(X(k,i,t-1) + X(k,i,t+1) - 2 \cdot X(k,i,t) \right)^2$ represents the squared error between coordinate $X(k,i,t)$ and the mid-point of coordinates $X(k,i,t-1)$ and $X(k,i,t+1)$.

We use the above simple choice of T for mobile sensor localization in our evaluation. A more sophisticated choice taking into account domain knowledge of the nature of sensor movement is likely to further improve the localization accuracy. We intentionally go with the simple choice to better illustrate the importance of taking temporal stability into account.

Incorporating low-rank constraints: Finally, to capture the low-rank nature of sensor mobility, we introduce a penalty term function

$$h(X, U, V) \triangleq \left\| M - U * V^T \right\|_F^2 \tag{5}$$

where $M = \mathsf{matricize}(X)$ is the $(d \cdot N) \times t_{\max}$ coordinate matrix obtained by collapsing the first two dimensions of X according to Eq. (1), U is a $(d \cdot N) \times r$ unknown factor matrix, and V is a $t_{\max} \times r$ unknown factor matrix, and r is the desired low rank. By keeping this penalty term small, we ensure that the coordinate matrix M has a good rank-r approximation: $M \approx U * V^T$.

Complete formulation: Putting everything together, we therefore try to find X, U, V that minimize the combined objective:

$$c(X, U, V) = \sum_t f(X, t) + \alpha \cdot g(X) + \beta \cdot h(X, U, V), \tag{6}$$

where α and β give the relative weights of temporal stability and low-rank constraints, respectively. We will show how to set α and β in Section 5.

Note that in Eq. (6), we can incorporate either or both of the penalty terms for temporal stability and low-rank structure (*i.e.*, $g(X)$ and $h(X, U, V)$). In this way, we can derive three different localization schemes: (i) Low Rank based Localization (LRL), which only exploits the low-rank structure in mobility (so $\alpha = 0$), (ii) Temporal Stability based Localization (TSL), which only leverages the temporal stability (so $\beta = 0$), and (iii) Temporal Stability and Low Rank based Localization (TSLRL), which simultaneously takes into account the temporal stability and the low-rank structure (so $\alpha \neq 0$ and $\beta \neq 0$). These localization schemes use distance estimation between neighbors and we call them *range-based* localization schemes. In addition, based on the general framework, we can also have *range-free* variants of these schemes, namely, LRL-RF, TSL-RF, and TSLRL-RF. These variants use only network connectivity for localization and can support nodes that do not have accurate distance estimation (*e.g.*, due to lack of RSS measurements).

A couple of comments follow. First, the penalty term $\sum_t f(X, t)$, is the fitting error over all time intervals. This assumes that we have distance measurements from all intervals t. Our evaluation uses this assumption. When the distance measurements from some intervals are missing (*e.g.*, due to measurement problems or the need to predict nodes' coordinates in a future interval), we can simply use the sum of $f(X, t)$ over the intervals that have distance measurements as the first term in the objective. Second, $f(X, t)$ is a quartic function (*i.e.*, function of the fourth degree) with respect to X, whereas $g(X)$ and $h(X, U, V)$ are quadratic functions (*i.e.*, functions of the second degree) with respect to X. So there is a mismatch between the units of the different penalty terms in Eq. (6). To make the formulation independent of the unit we use for X, we can first normalize X such that the communication range equals 1. We assume the use of such normalization in the rest of the paper.

5. OPTIMIZATION

Optimization algorithm: We apply a quasi-Newton optimization algorithm L-BFGS [28] to find a local optimum of the above nonlinear objective function $c(X, U, V)$. Quasi-Newton methods [58] are variants of the well-known Newton's method [57] for finding a stationary point (*i.e.*, a point where the gradient is 0) of a given objective function. Newton's method assumes that the objective function can be locally approximated as a quadratic function in the region around the optimum, and uses the first and second derivatives (*i.e.*, gradient and Hessian) to find the stationary point. However, it is often expensive to directly compute the entire Hessian matrix for large optimization problems. To achieve better scalability, quasi-Newton methods do not compute the Hessian matrix directly. Instead, they update the Hessian matrix by analyzing successive gradient vectors.

L-BFGS stands for "limited memory BFGS". It is a particular quasi-Newton method that uses the Broyden-Fletcher-Goldfarb-Shanno (BFGS) update to approximate the Hessian matrix. L-BFGS only maintains a short history of the most recent m updates of the position (X, U, V) and gradient $\nabla c(X, U, V)$, where the history length m is typically less than 10. As a result, L-BFGS is particularly suited for solving optimization problems with a large number of variables.

We currently use the L-BFGS implementation in the *minFunc* package [33]. We set the history length $m = 5$, which provides good efficiency without compromising accuracy.

Choosing a good initial solution: Since L-BFGS is a gradient-based local search algorithm, it is critical to have a good initial solution. Without a good initial solution, the optimization tends to get stuck at local optima with poor localization accuracy, especially when d is small (*e.g.*, $d = 2$ or $d = 3$).

After extensive experiments, we find that the following initialization strategy often results in significant accuracy improvement:

1. First solve the problem in d'-dimensional Euclidean space using a random initial solution, where $d' \geq d$ (*e.g.*, $d' = 4$ in our experiments). Let the solution be X''.
2. Compute the pair-wise distance matrix using X''.
3. Use multi-dimensional scaling (MDS) [12] to map the distance matrix obtained in step 2 into a solution in d-dimensional Euclidean space. Let the solution be X'.
4. Use X' as the initial solution and solve the original problem in d-dimensional Euclidean space.

Note that each point in the d-dimensional Euclidean space can be embedded into a d'-dimensional Euclidean space by setting its coordinates in the additional dimensions to 0. However, the converse is not true. That is, a point in a d'-dimensional space may not reside in a d dimensional space. Thus the d'-dimensional solution obtained in step 1 does not directly provide a solution for the original d-dimensional problem. That is why we apply MDS to project the solution back to the d-dimensional Euclidean space and use the result as an initial solution. Note that unlike existing MDS techniques in sensor localization [46, 47], we can directly apply the classic MDS [12], because the pair-wise distance matrix obtained in step 2 contains no missing entries.

The intuition behind the above initialization procedure is the following. When the dimension d is low, the feasible solution space for X may get partitioned into isolated islands. As a result, local search can easily get stuck at local optima with poor localization accuracy. In a higher dimensional Euclidean space, due to the higher degree of freedom, different regions of the feasible solution space become better connected and it becomes harder to get stuck at a local optimum. As a result, a random starting point often suffices to yield a local optimum with high localization accuracy in the higher dimensional Euclidean space. Applying MDS to project this solution back to the low dimensional space can therefore yield a good initial solution for the original problem.

Our evaluation uses a single initial solution. One could also use multiple starting points and choose the one that yields the lowest error. This may further improve the localization accuracy at the expense of larger running time.

Tuning parameters α, β, and r: α and β control the importance of temporal stability and low-rank constraints, respectively. Their values depend on how noisy distance measurements are and how stable and low-rank the coordinate matrix is. When there is significant measurement noise, their values should increase to avoid being dominated by the large fitting error term. Moreover, α should also increase when the coordinate matrix exhibits strong temporal stability, and decrease otherwise. Therefore α depends on $ratio = \frac{\sum_t f(X,t)}{g(X)}$. To automatically adapt to diverse scenarios, we choose α using the following simple two iterations: we set $\alpha = 1$ in the first iteration and solve the optimization problem in Eq. 6; in the second iteration, if $ratio > 1$, we set $\alpha = min(ratio, 10)$. We bound α by 10 to prevent it from being too large. Similar adaptation could be used in choosing β. For simplicity, our evaluation uses a simple setting of $\beta = 0.1$, since we find it is more important to adapt α due to the higher importance of temporal stability constraints. Another parameter required in low rank constraints is the rank r. Our evaluation uses $r = 3$. In the future, we plan to explore methods for automatically determining the appropriate rank r based on partial distance matrices.

Time complexity: As an iterative algorithm, the time complexity of L-BFGS depends on both the amount of time spent during each iteration and the number of iterations.

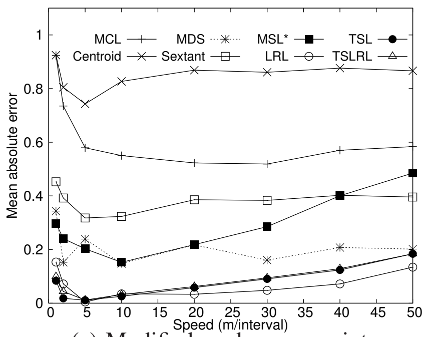
- *The time spent in each iteration.* In our context, the time spent during each iteration is dominated by the time for computing $\sum_t f(X,t)$ and $\sum_t \nabla f(X,t)$. As shown in Eq. (2), the expression for each $f(X,t)$ comprises $O((N+N_{\text{anch}})\cdot N)$ terms, each involving $O(d)$ variables. So it takes $O((N + N_{\text{anch}}) \cdot N \cdot d)$ time to compute $f(X,t)$ and $\nabla f(X,t)$. So the total time complexity during each iteration is $O((N+N_{\text{anch}})\cdot N\cdot d\cdot t_{\max})$.
- *The number of iterations.* In our implementation, we simply run the L-BFGS algorithm for a fixed number of iterations (denoted as N_{iter}). Currently, we conservatively set $N_{\text{iter}} = 1000$, which ensures convergence in all our simulations and experiments. In our future work, we plan to incorporate less conservative convergence tests that allow the L-BFGS algorithm to terminate early.

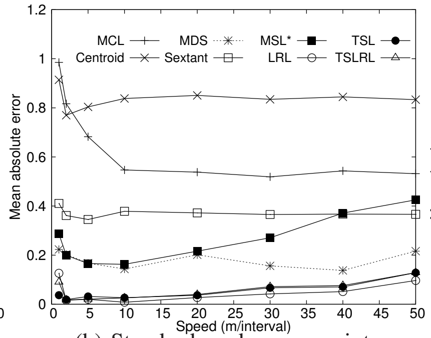
Putting everything together, the time complexity for our current solution is given by $O((N + N_{\text{anch}}) \cdot N \cdot d \cdot t_{\max} \cdot N_{\text{iter}})$. The amortized cost per time interval is $O((N + N_{\text{anch}}) \cdot N \cdot d \cdot N_{\text{iter}})$.

6. SIMULATION

6.1 Simulation Methodology

We use a publicly available network simulator [32] for our evaluation. As in [31, 42], we quantify node density as the average number of nodes, including both regular nodes and anchor nodes, in one hop transmission range. It can be calculated as $\frac{\pi R^2 (N+N_{\text{anch}})}{area}$, where R is communication range, $area$ is the size of the deployment area, and N and N_{anch} are the total numbers of regular nodes and anchor nodes, respectively. Similarly, we quantify anchor density as the average number of anchors in one hop transmission range, calculated as $\frac{\pi R^2 N_{\text{anch}}}{area}$. Unless otherwise specified, we randomly place 50 nodes (including 5 anchor nodes) in a $200m \times 200m$ area, and use the following default parameter setting according to [30, 31]: node density of 10, anchor density of 1, communication range of $50m$, and a maximum speed of 10 $m/interval$. Furthermore, since MCL and MSL* require a warm-up period, we use 30 intervals as input to the evaluation and quantify the localiza-


(a) Modified random waypoint


(b) Standard random waypoint

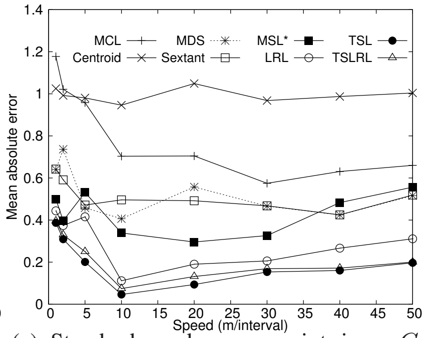

(c) Standard random waypoint in a C-shaped area

Figure 3: Comparison of different localization schemes using random waypoint models.

tion error during the last 10 intervals. Without such a warm-up period, the benefit of our localization schemes over MCL and MSL* is even higher. For each configuration, we conduct 10 random runs. We further vary each of these parameters to understand its impact.

Our evaluation uses both synthetic and real mobility traces. We generate synthetic node movement using the modified random waypoint model [22] (as described in Section 3) to overcome the limitation of speed reduction over time. We also try the standard random waypoint model and observe similar performance. Therefore for most evaluation, we report the results from the former mobility model in the interest of space. In addition, we also use the real mobility traces summarized in Table 1. In order to run 30 intervals across 10 random runs, we extract 300 consecutive intervals from these traces, except that ZebraNet has only 90 intervals, which allow us to conduct only 3 random runs.

We compare our localization schemes with the following existing localization schemes: (i) Centroid [8], (ii) Multidimensional Scaling (MDS) [47], (iii) Sextant [19], (iv) MCL [22], and (v) MSL* [42]. The first three are well known localization schemes for static wireless networks and the last two are state-of-the-art localization schemes for mobile wireless networks. Refer to Section 2 for the description of these localization approaches.

We quantify the localization error using the mean absolute error (MAE), which has been widely used in previous studies. It is computed as $\mathsf{mean}_{i,t}(dist(X(:,i,t), X'(:,i,t))$, where X and X' denote the actual and estimated coordinates, respectively, and $dist(X(:,i,t), X'(:,i,t))$ is the Euclidean distance between the actual and estimated coordinates for node i at time t.

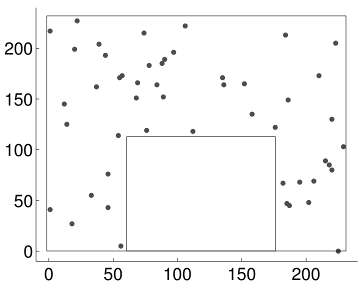


Figure 4: The C-shaped area used in simulation.

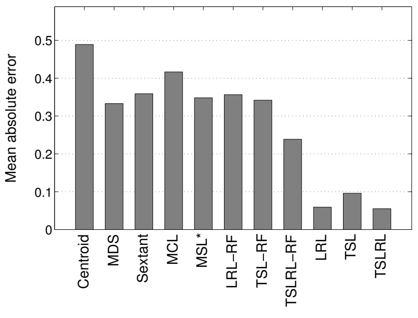
6.2 Simulation Results

Varying mobility: Figure 3(a) and (b) compare different localization schemes using the modified and standard random waypoint models, respectively. We make the following observations. First, in all cases, all our localization schemes significantly out-perform the other schemes. For example, TSLRL reduces the MAE by a factor of 4.7-63.5 over Centroid, 1.1-20.4 over MDS, 2.2-27.2 over Sextant, 3.2-49.5 over MCL, and 2.6-17.3 over MSL*. Second, the localization error in the modified mobility model is slightly higher than that in the standard mobility model due to more frequent change in velocity, but their general trends and relative performance across different schemes are very similar. Therefore we use the modified random waypoint for the rest of our evaluation since it is used in [31, 42]. Third, the errors of all mobile network localization schemes initially decrease as we increase mobility because mobility allows us to obtain additional constraints across different time intervals. As we further increase mobility, the error starts to increase, because the increased mobility reduces the extent of the low-rank structure and temporal stability. This behavior is more prominent in the modified random waypoint model, as the node selects a new speed every interval, whereas in the standard random waypoint model the speed remains constant for consecutive intervals until the node reaches its destination.

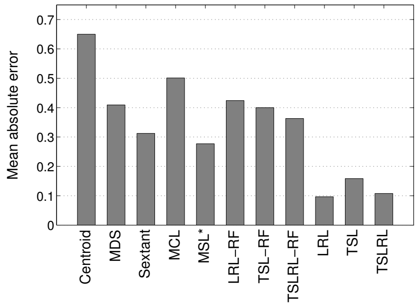
We also evaluate using an irregular C-shaped area shown in Figure 4. It has a $230m \times 230m$ square with a $115m \times 115m$ block taken off. We generate nodes' coordinates over time using the standard random waypoint mobility within the area. To ensure every node's position falls within the C-shaped region, whenever it is about to traverse outside the region, we re-select a new destination and move towards it at a randomly selected speed. As shown in Figure 3(c), the results are similar to before, except that the accuracy of MDS degrades and sometimes under-performs MSL* and Sextant. This is a well known issue with MDS, because in irregular topologies the true Euclidean distance between two nodes does not correlate well with the shortest path distance, which is used as the input to MDS. In comparison, our localization schemes only use the distance information between direct neighbors to generate equality constraints and are less sensitive to the above issue. As a result, they continue to perform the best.

We further evaluate using real mobility traces. We pick 100 vehicles randomly from the vehicular traces and use all the nodes from the KAIST campus traces and ZebraNet traces. For all the traces, we scale down the distance to get an approximate node density of 10 and anchor density of 1. Figure 5 summarizes the results. As we can see, our range-based localization schemes consistently yield much lower errors than the other schemes across all the traces. The range-free versions, namely, LRL-RF, TSL-RF, and TSLRL-RF, perform worse than their corresponding range-based counterparts due to lack of distance estimation. Among the existing schemes, MDS, Sextant, and MSL* perform better. However, their accuracy is still much worse than our range-based schemes since they do not fully exploit topology and mobility information.

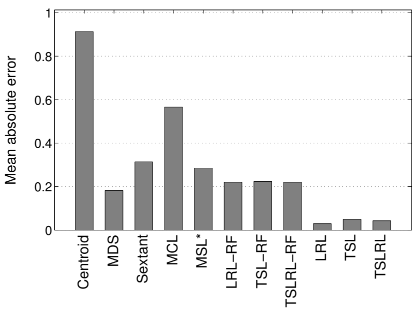
Varying node density: Next we evaluate the impact of node density by fixing the area to $200m \times 200m$ as before and varying the number of total nodes from 10 to 70. As shown in Figure 6, our lo-



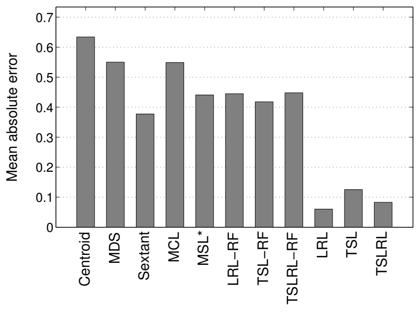
(a) San Francisco taxis



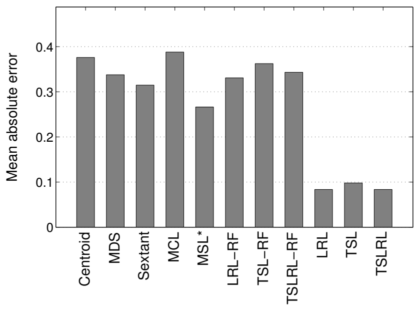
(b) Seattle bus



(c) Shanghai taxis



(d) ZebraNet



(e) Human mobility

Figure 5: Comparison of different localization schemes using real traces.

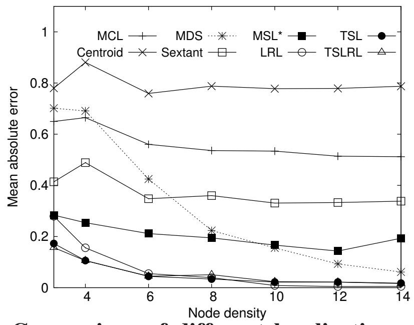


Figure 6: Comparison of different localization schemes with varying node density.

calization schemes continue to perform the best under all node density. In addition, all the localization schemes have lower error with increasing node density. The only exception is Centroid, which sees similar error over all node density. This is because Centroid extracts location information only from anchor nodes and does not exploit the distance information between regular nodes. Therefore its accuracy is determined only by the anchor nodes and benefits little from higher density of regular nodes. Similar effects were reported in [22].

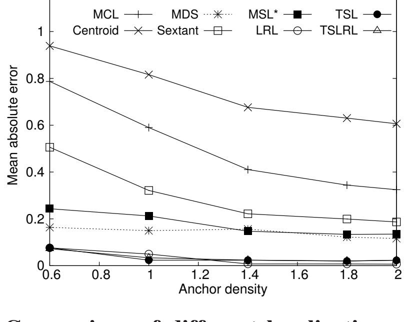
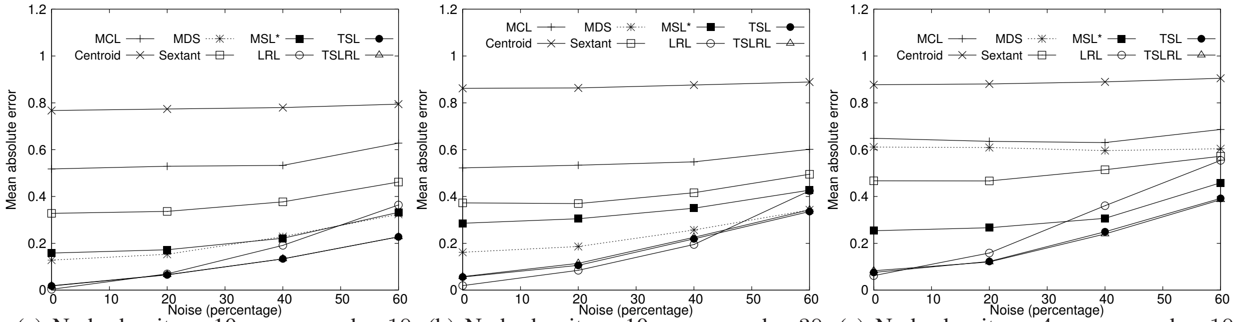


Figure 7: Comparison of different localization schemes with varying anchor density.

Effect of anchor density: We further vary the anchor density by fixing the area to the default size as before and changing the number of anchor nodes from 3 to 10. Figure 7 summarizes the results. As we would expect, all the localization schemes benefit from increasing anchor density. Moreover, all three of our localization schemes significantly out-perform the other schemes.

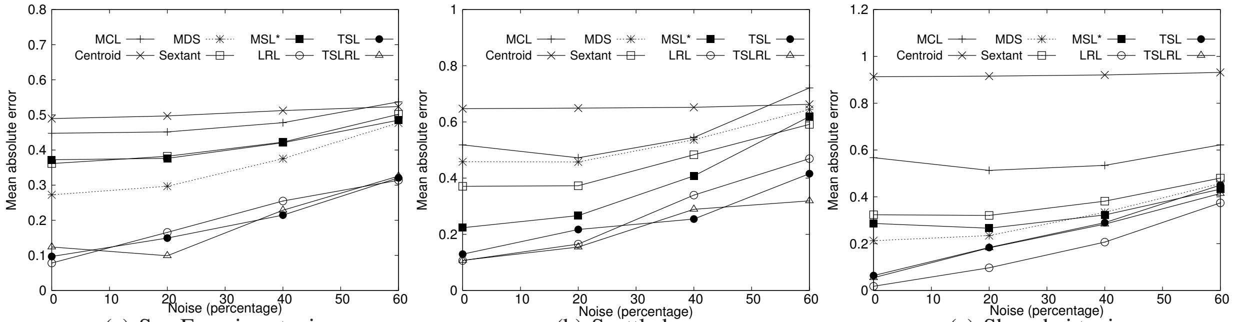
Varying the amount of noise: It is important to study the accuracy of all the schemes under measurement noise, since noise is common in real deployment. Measurement noise comes from two factors: (i) there is irregularity in transmission range, *i.e.*, not all nodes have the same transmission range and even for the same node its transmission range is not the same in all directions, and (ii) a neighboring node may estimate inaccurate distance based on its RSS measurements. To capture the first effect, we generate a random number r_{ij} uniformly distributed from $[1-noise, 1+noise]$ and we vary $noise$ from 0 to 60%. The new transmission range between the node pair becomes $r_{ij} \cdot R$. To capture the second effect, we perturb the distance estimation as d_{ij}/r_{ij}, where d_{ij} is the actual distance between i and j and r_{ij} is the scaling factor used to generate noisy transmission range between i and j. In this way, we ensure the error added to the transmission range is consistent to the error added to the distance estimation. We then drop the entry (i, j) in the distance matrix if the actual distance between i and j is larger than the salted transmission range (since they are not direct neighbors and cannot measure RSS), and input this distance matrix to all the localization schemes. Moreover, since the localization schemes are not aware of the injected noise, they consider R as the actual communication range (*e.g.*, our localization schemes use R to generate lower bounds and upper bounds).

Figure 8 compares various localization schemes by varying the amount of noise. Figure 8(a) shows the results under the default network configuration, Figure 8(b) shows the results under higher mobility, and Figure 8 (c) shows the results under lower node density. We make the following observations. First, as we would expect, increasing noise degrades the accuracy of all the localization schemes. Among them, the error in Centroid increases slowest with increasing noise, because it estimates its location as the center of all



(a) Node density = 10, max speed = 10 $m/interval$ (b) Node density = 10, max speed = 30 $m/interval$ (c) Node density = 4, max speed = 10 $m/interval$

Figure 8: Varying noise in 2-D networks.



(a) San Francisco taxis (b) Seattle bus (c) Shanghai taxis

Figure 9: Varying noise in the real vehicular traces.

anchor nodes it hears from and is not affected significantly by measurement errors. Similar effects were observed in [22]. Second, TSL and TSLRL continue to yield the lowest errors under all noise values. They reduce the MAE by a factor of 2.3-40.9 over Centroid, 1-7.4 over MDS, 1.4-6.8 over Sextant, 1.8-27.6 over MCL, and 1.2-8.4 over MSL*. Third, LRL initially performs similarly to TSL and TSLRL and then slightly under-performs as noise increases. This is likely because increasing noise may affect low rank structure and reduce the effectiveness of low rank constraints. In comparison, temporal stability is more robust to noise. Fourth, increasing the maximum speed from 10 $m/interval$ to 30 $m/interval$ slightly degrades the accuracy of various schemes. Among them, MSL* is affected the most, because it uses the maximum speed to generate feasible node positions during the next intervals and location uncertainty increases with mobility. Finally, as we would expect, the accuracy of all the schemes degrades as the node density decreases due to fewer location constraints.

Figure 9 evaluates the impact of noise on real vehicular traces. As in the synthetic traces, increasing noise degrades the accuracy of most localization schemes. Moreover, our localization schemes continue to out-perform the other schemes under all noise values.

To demonstrate the flexibility of our scheme, we also consider varying noise when nodes have 3-D coordinates. We place 50 nodes including 5 anchor nodes randomly in a cube with each side of $140m$. Figure 10 compares our localization schemes with the other schemes except Sextant, which works only in 2-D. We observe that TSL and TSLRL perform similarly and their curves overlap. They both yield the lowest error in all the cases. LRL performs slightly worse as noise increases. Moreover, Centroid yields the highest error since it only uses anchor nodes for localization, which gives very limited location constraints.

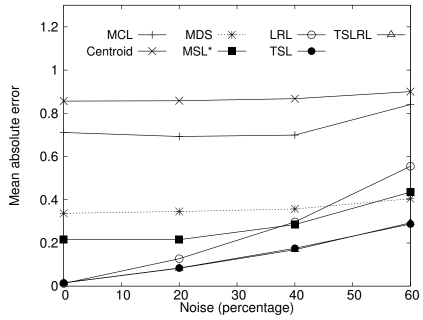


Figure 10: Varying noise in 3-D networks.

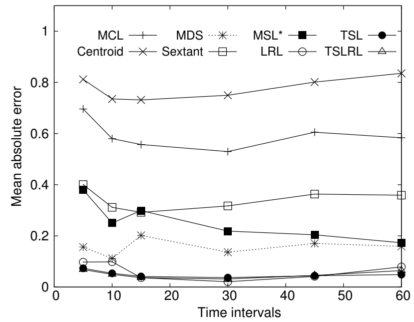


Figure 11: Impact of the number of time intervals.

Varying the number of time intervals: Figure 11 shows the localization error as we vary the total number of time intervals used for localization. We use the first 5 time intervals as the warm-up period required by MCL and MSL*, and vary the number of time intervals for localization from 5 to 60. As we can see, TSL, LRL and TSLRL consistently yield the lowest errors. In comparison, MCL incurs MAE of 0.58 even with 60 intervals. MSL* improves the convergence over MCL by letting nodes use information only from the neighbors that have more accurate coordinates, and reduces mean absolute error to 0.2-0.4. However, it still performs

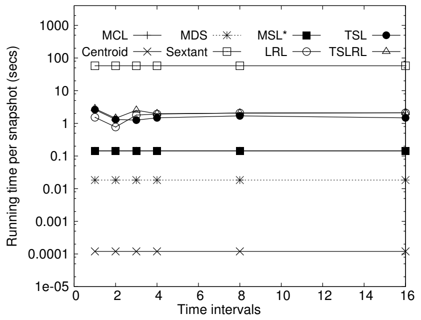


Figure 12: Running Time

much worse than our schemes because it does not fully exploit location constraints during each time interval nor the structure in mobility, other than the maximum speed.

Running time: Finally we compare the running time of different approaches on a Linux machine with Intel(R) Core(TM)2 Duo CPU E8200 2.66GHz processor, 2GB memory, and 6 MB cache. Figure 12 shows the average running time per interval as we vary the number of intervals for the default configuration (*i.e.*, 50 nodes including 5 anchor nodes, node density of 10, anchor density of 1, and a maximum speed of 10 $m/interval$). Our schemes (*i.e.*, LRL, TSL, TSLRL) and MDS are implemented in Matlab, while the other schemes were implemented in Java by other researchers. Java implementation is generally faster than Matlab. As shown in Figure 12, all of the schemes have close to constant running time per interval. In other words, their total running time increases linearly with the number of intervals. The ranking of their running time is as follows: Centroid $<$ MDS $<$ MSL* $\approx$ MCL $<$ LRL $\approx$ TSL $\approx$ TSLRL $<$ Sextant. Sextant has highest running time: around 1 minute per interval, because computing the regions that satisfy all the measurement constraints is expensive. Our approaches (*i.e.*, LRL, TSL, and TSLRL) take 1-2 seconds per interval. This is longer than the existing schemes except Sextant because we try to fully exploit the location information within each interval and mobility structure across intervals. However this running time is still sufficient for practical use. Moreover, our implementation has considerable room for optimization (*e.g.*, adaptively choosing the number of iterations as mentioned in Section 5 and converting Matlab to C implementation), which we plan to explore as part of our future work.

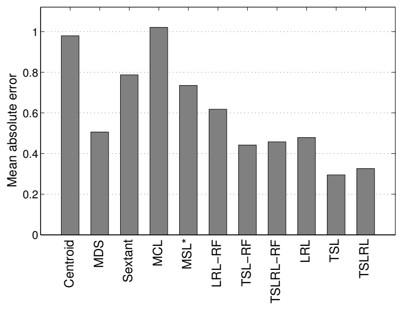
Summary: Our simulation results show that our localization schemes significantly out-perform the existing schemes under a wide range of scenarios. They are highly robust to measurement errors. They are also sufficiently efficient for practical use.

7. TESTBED EXPERIMENTS

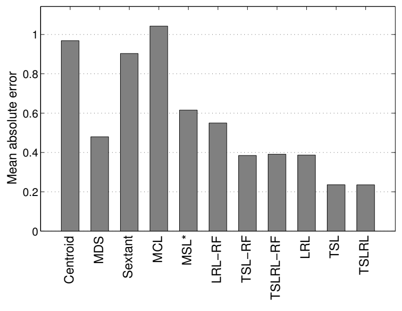
7.1 Experimental Methodology

In this section, we evaluate our localization scheme in a sensor testbed. We deploy a testbed consisting of either 25 or 36 $mica2$ motes with 915MHz radios on a single floor of an office building. This allows us to evaluate our localization algorithm under realistic radio characteristics. In our first experiment, we randomly place them in a square with $5.5m \times 5.5m$. This gives node density of 20. We use the standard random waypoint [22] to generate the motes' coordinates at different time intervals. We use two types of mobility: low mobility with a maximum speed of $0.2R$ per time interval and high mobility with a maximum speed of $0.6R$ per time interval. In our second experiment, we place the motes in a C-shaped topology shown in Figure 15(a), occupying a total area of $6.5m \times 6.5m$ with a smaller square of $3.4m \times 3.4m$ taken away. It has a node density of 15. We again use the standard random waypoint to generate the coordinates in the subsequent time intervals while ensuring that the motes move within the region. For diversity, we use medium mobility with a maximum speed of $0.4R$. In both topologies, we use 15 time intervals and move these motes by hand at the beginning of each interval to ensure the actual locations of the motes correspond to the generated locations.

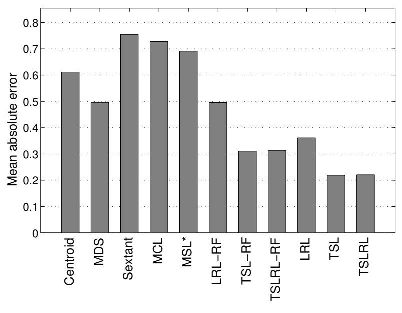
Among these motes, we randomly choose 6 anchor nodes in both topologies. We adjust the transmission power to $-24dBm$ for all control and data traffic. This gives the communication range of



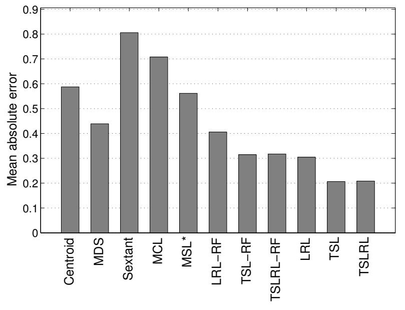
(a) 25 nodes, low mobility



(b) 25 nodes, high mobility

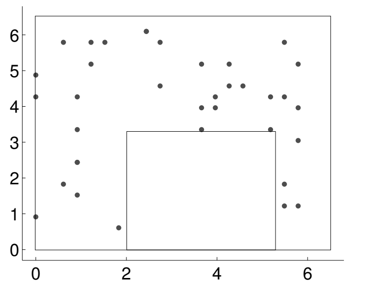


(c) 36 nodes, low mobility

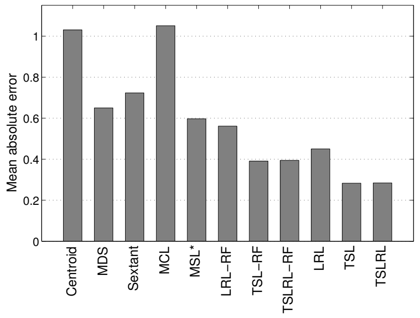


(d) 36 nodes, high mobility

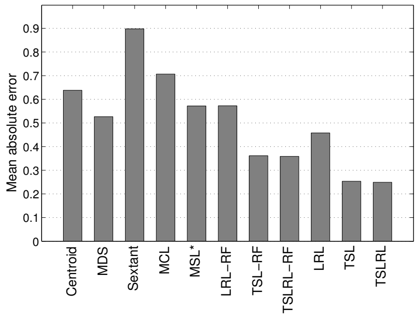
Figure 13: Comparison of various location schemes in the testbed when nodes are located in a square.



(a) C-shaped topology used in experiments



(b) 25 nodes, irregular topologies



(c) 36 nodes, irregular topologies

Figure 15: Comparison of various location schemes in the testbed when nodes are located in a C-shaped region.

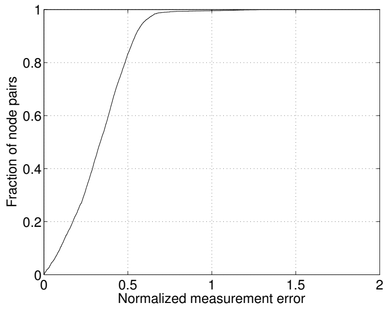


Figure 14: CDF of measurement error in the testbed.

about $2.3m$, and yields multi-hop topologies with up to 3 hops. One mote, referred as the sink, is attached to the MIB600 Ethernet board, which is connected to a power outlet. We use the sink to log all measurement data. The other motes are powered by batteries.

During each time interval, the sink first broadcasts a route establishment frame (REF), which is flooded throughout the network so that every other node can route towards the sink by reversing the shortest path along which REF traversed. To improve the reliability of the route, every node selects the path with the smallest ETX to route towards the sink. ETX quantifies the total number of expected transmissions from a source to a destination [13]. To support such a route selection, an intermediate node i appends the ETX between the sink and node i to REF before forwarding.

Next every mote broadcasts several packets at a random time so that its neighbors can measure the received signal strength (RSS) from it. Each mote broadcasts once every 1.5 seconds. The starting time of each broadcast packet is randomized to minimize collisions with other broadcast packets. Every mote periodically sends a report to the sink to summarize the measured received signal strength from all nodes it hears during the current period. The report is routed using the shortest ETX path selected above. We then take measurements collected from the sink over 15 intervals and use them to evaluate the performance of various localization schemes. We estimate the distance between neighboring nodes during a given time interval using average RSS of all the packets received during that time interval.

7.2 Experimental Results

Figure 13 shows the mean absolute error for 25-node and 36-node networks in the first topology. We make the following observations. First, all our range-based localization schemes significantly out-perform the other schemes. In particular, TSL and TSLRL reduce the MAE by a factor of 2.8-4.1 over Centroid, 1.6-2.3 over MDS, 2.4-3.9 over Sextant, 3.1-4.4 over MCL, and 2.3-3.2 over MSL*. Second, our range-free versions also perform well: TSL-RF and TSLRL-RF both yield lower errors than the existing schemes. LRL-RF occasionally performs worse than MDS, the best of the existing schemes, because it only exploits low rank structure, which may be affected by measurement noise in the testbed. Third, the gap between the range-free and range-based schemes is smaller than in simulation. This is because the testbed experiments have significant errors in estimating the distances based on RSS measurements. This is further confirmed by Figure 14, which plots the CDF of the normalized error in RSS-based distance estimation (*i.e.*, the difference between the estimated and actual distances normalized by the actual distance). Such measurement noise makes the equality constraints noisy. Finally, comparing the five existing localization schemes, MDS performs the best among the static localization schemes, and MSL* significantly improves accuracy of MCL in mobile network localization due to its faster convergence. It is interesting to note that even though MSL* leverages mobility information, it still does not perform as well as MDS because it does not fully take advantage of network topology during each time interval. By exploiting both the topology information at individual intervals and the structural properties in mobility, our approaches achieve much better accuracy.

Figure 15 (b) and (c) further plot the localization errors in 25-node and 36-node networks deployed in a C-shaped region, as shown in Figure 15(a). Both range-free and range-based versions of our localization schemes out-perform the existing schemes. The ranged-based of TSL and TSLRL continue to perform the best: they reduce the MAE by a factor of 2.5-3.6 over Centroid, 2.1-2.3 over MDS, 2.5-3.6 over Sextant, 2.8-3.7 over MCL, and 2.1-2.3 over MSL*. In addition, as we would expect, the accuracy of MDS degrades in irregular topologies due to lack of strong correlation between the shortest path distance and Euclidean distance, as described in Section 6.2.

Summary: Our testbed experiments show that our localization schemes out-perform the existing schemes in a range of settings. Among them, the range-based versions of TSL and TSLRL consistently perform the best due to the effectiveness and robustness of temporal stability constraints.

8. CONCLUSION

In this paper, we analyze both real and synthetic mobility traces and show that they all exhibit temporal stability and low-rank structure. Motivated by these observations, we develop novel localization schemes for mobile networks. Our schemes explicitly exploit these structural properties in mobility while leveraging network topology information during each time interval. Using extensive simulation based on real and synthetic mobility traces, we show that our localization schemes significantly out-perform the existing schemes. Our testbed experiments further confirm the effectiveness of our approaches.

Acknowledgments

The research is supported in part by NSF Grants CNS-0546755, CNS-0916309, and CNS-0916106. We are grateful to Moustafa Youssef and anonymous reviewers for their valuable comments. We thank David Evans, Saikat Guha and Marcelo Martins for sharing their code with us.

9. REFERENCES

[1] Alert systems. `http://www.alertsystems.org/`.

[2] A. Baggio and K. Langendoen. Monte Carlo localization for mobile wireless sensor networks. *Ad Hoc Networks*, 6(5):718–733, 2008.

[3] P. Bahl and V. N. Padmanabhan. RADAR: an in-building RF-based user location and tracking system. In *Proc. of IEEE INFOCOM*, Mar. 2000.

[4] L. Benmohamed, P. Chimento, B. Doshi, B. Henrick, and I. J. Wang. Sensor network design for underwater surveillance. In *Proc. of Military Communications Conference (MILCOM)*, pages 1–7, 2006.

[5] P. Biswas, T. C. Liang, K. C. Toh, T. C. Wang and Y. Ye. Semidefinite programming approaches for sensor network localization with noisy distance measurements. *IEEE Transactions on Automation Science and Engineering*, 2006.

[6] P. Biswas and Y. Ye. Semidefinite programming for ad hoc wireless sensor network localization. In *Proc. of 3rd IPSN*, 2004.

[7] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *Proc. of ACM MobiCom*, Oct. 1998.

[8] 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, Oct. 2000.

[9] Cabspotting project. `http://www.cabspotting.com`.

[10] E. Candes and T. Tao. Near optimal signal recovery from random projections: Universal encoding strategies. *IEEE Transactions on Information Theory*, 2006.

[11] A. Cerpa, J. Elson, D. Estrin, L. Girod, M. Hamilton, and J. Zhao. Habitat monitoring: application driver for wireless communications technology. In *Proc. of ACM SIGCOMM Workshop on Data Communications*, Aug. 2001.

[12] Multidimensional scaling. `http://en.wikipedia.org/wiki/Multidimensional_scaling`.

[13] D. D. Couto, D. Aguayo, J. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. In *Proc. of ACM MobiCom*, Sept. 2003.

[14] Disaster recovery. `http://www.wireless-center.net/WLANs-WPANs/1347.html`.

[15] D. Donoho. For most large under-determined systems of linear equations, the minimal L1-norm near-solution approximates the sparsest near-solution. `http://www-stat.stanford.edu/~donoho/Reports/2004/l1l0approx.pdf`.

[16] D. Donoho. For most large under-determined systems of linear equations, the minimal L1-norm solution is also the sparsest solution. `http://www-stat.stanford.edu/~donoho/Reports/2004/l1l0EquivCorrected.pdf`.

[17] D. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 2006.

[18] Global positioning system standard positioning service specification. *United States Coast Guard Navigation Center*, June 1995.

[19] S. Guha, R. N. Murty, and E. G. Sirer. Sextant: A unified framework for node and event localization in sensor networks. In *Proc. of ACM MobiHoc*, May 2005.

[20] T. He, S. Krishnamurthy, J. A. Stankovic, T. Abdelzaher, L. Luo, R. Stoleru, T. Yan, L. Gu, J. Hui, and B. Krogh. Energy-efficient surveillance system using wireless sensor networks. In *Proc. of ACM MobiSys*, pages 270–283, 2004.

[21] J. Hightower and G. Borriello. Location systems for ubiquitous computing. *IEEE Computer*, Aug 2001.

[22] L. Hu and D. Evans. Localization for mobile sensor networks. In *Proc. of ACM MobiCom*, Sept. 2004.

[23] Human mobility traces. `http://crawdad.cs.dartmouth.edu/ncsu/mobilitymodels`.

[24] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. S. Peh, and D. Rubenstein. Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with ZebraNet. *SIGARCH Comput. Archit. News*, 30(5):96–107, 2002.

[25] B. Karp and H. Kung. Greedy perimeter stateless routing for wireless networks. In *Proc. of ACM MobiCom*, Aug. 2000.

[26] M. Korkea-aho. Context-aware applications survey. *Internetworking Seminar*, Apr. 2000. `http://users.tkk.fi/~mkorkeaa/doc/context-aware.html`.

[27] U. Kubach and K. Rothermel. Exploiting location information for infostation-based hoarding. In *Proc. of ACM MobiCom*, Jul. 2001.

[28] D. C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Math. Program.*, 45(3):503–528, 1989.

[29] D. Madigan, E. Elnahrawy, and R. P. Martin. Bayesian indoor positioning systems. In *Proc. of IEEE INFOCOM*, Mar. 2005.

[30] M. H. T. Martins, H. Chen, and K. Sezaki. OTMCL: Orientation tracking-based Monte Carlo localization for mobile sensor networks. In *Proc. of the Six International Conference on Networked Sensing Systems (INSS)*, 2009.

[31] Mesh connectivity layer. `http://research.microsoft.com/mesh/#software`.

[32] Localization for mobile sensor networks. `http://www.cs.virginia.edu/mcl`.

[33] minFunc. `http://www.cs.ubc.ca/~schmidtm/Software/minFunc.html`.

[34] Mobile sensing group at Dartmouth. `http://sensorlab.cs.dartmouth.edu/`.

[35] Mobile sensing. `http://www.mobilesensing.org/`.

[36] D. Moore, J. Leonard, D. Rus, and S. Teller. Robust distributed network localization with noisy range measurements. In *Proc. of ACM SenSys*, Nov. 2004.

[37] D. Niculescu and B. Nath. VOR base stations for indoor 802.11 positioning. In *Proc. of ACM MobiCom*, Sept. 2004.

[38] U. of Texas at Arlington. Mobile sensor networks. `http://arri.uta.edu/smart_micromachines/distributed_devices/mobile_sensor_networks.html`.

[39] V. N. Padmanabhan, L. Qiu, and H. Wang. Server-based inference of Internet performance. In *Proc. of IEEE INFOCOM*, Mar. 2003.

[40] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. In *Proc. of ACM MobiCom*, Aug 2000.

[41] A. Rao, S. Ratnasamy, C. Papadimitriou, S. Shenker, and I. Stoica. Geographic routing without location information. In *Proc. of ACM MobiCom*, Sept. 2003. `http://www.cs.ucla.edu/classes/fall03/cs218/paper/p96-rao.pdf`.

[42] M. Rudafshani and S. Datta. Localization in wireless sensor networks. In *Proc. of IPSN*, pages 51–60, 2007.

[43] A. Savvides, C. Han, and M. B. Strivastava. Dynamic fine-grained localization in ad-hoc networks of sensors. In *Proc. of ACM MobiCom*, Jul. 2001.

[44] A. Savvides, H. Park, and M. B. Srivastava. The bits and flops of the n-hop multilateration primitive for node localization problems. In *Proc. of WSNA'02*, Sep. 2002.

[45] Seattle bus traces. `http://crawdad.cs.dartmouth.edu/rice/ad_hoc_city`.

[46] Y. Shang and W. Ruml. Improved MDS-based localization. In *Proc. of IEEE INFOCOM*, Apr. 2004. `http://www.ieee-infocom.org/2004/Papers/55_1.PDF`.

[47] Y. Shang, W. Ruml, Y. Zhang, and M. Fromherz. Localization from mere connectivity. In *Proc. of ACM MobiHoc*, Jun. 2003. `http://www.sigmobile.org/mobihoc/2003/papers/p201-shang.pdf`.

[48] C. Sharp, S. Schaffert, A. Woo, N. Sastry, C. Karlof, S. Sastry, and D. Culler. Design and implementation of a sensor network system for vehicle tracking and autonomous interception. *Proc. of the Second European Workshop on Wireless Sensor Networks*, pages 93–107, 2005.

[49] D. C. Steere, A. Baptista, D. McNamee, C. Pu, and J. Walpole. Research challenges in environmental observation and forecasting systems. In *Proc. of 6th International Conference on Mobile Computing and Networking*, 2000.

[50] E. Stevens-Navarro, V. Vivekanandan, and V. W. S. Wong. Dual and mixture Monte Carlo localization algorithms for mobile wireless sensor networks. In *Proc. of IEEE Wireless Communications and Networking Conference (WCNC)*, pages 4024–4028, 2007.

[51] Underwater acoustic sensor networks. `http://www.ece.gatech.edu/research/labs/bwn/UWASN/`.

[52] Volcano monitoring (Harvard Sensor Networks Lab). `http://fiji.eecs.harvard.edu/Volcano`.

[53] F. Wang, L. Qiu, and S. Lam. Probabilistic region-based localization for wireless networks. *ACM Mobile Computing and Communications Review (MC2R) Special Issue on Localization*, Jan. 2007.

[54] Z. Wang, S. Zheng, S. Boyd, and Y. Ye. Further relaxations of the SDP approach to sensor network localization. `http://www.stanford.edu/~yyye/relaxationsdp9.pdf`.

[55] R. Want, A. Hopper, V. Falcao, and J. Gibbons. The active badge location system. *ACM Transactions on Information Systems*, Jan. 1992.

[56] Wikipedia. Nearest-neighbor interpolation. `http://en.wikipedia.org/wiki/Nearest-neighbor_interpolation`.

[57] Wikipedia. Newton's method in optimization. `http://en.wikipedia.org/wiki/Newton's_method_in_optimization`.

[58] Wikipedia. Quasi-Newton method. `http://en.wikipedia.org/wiki/Quasi-Newton_method`.

[59] J. Yoon, M. Liu, and B. Noble. Sound mobility models. In *Proc. of ACM MobiCom*, Sept. 2003.

[60] ZebraNet traces. `http://crawdad.cs.dartmouth.edu/princeton/zebranet`.

[61] Y. Zhang, M. Roughan, W. Willinger, and L. Qiu. Spatio-temporal compressive sensing and Internet traffic matrices. In *Proc. of ACM SIGCOMM*, pages 267–278, Aug. 2009.

[62] H. Zhu, M. Li, Y. Zhu, and L. M. Ni. Hero: Online real-time vehicle tracking. *IEEE Transactions on Parallel and Distributed Systems*, 20(5):740–752, 2009.