Tracking Mobile Nodes Using RF Doppler Shifts

Branislav Kusy, Akos Ledeczi, Xenofon Koutsoukos
Vanderbilt University, Nashville, TN, USA
branislav.kusy@gmail.com

Abstract

In this paper, we address the problem of tracking cooperative mobile nodes in wireless sensor networks. Aiming at a resource efficient solution, we advocate the use of sensors that maintain their location information and rely on the tracking service only when their locations change. In the proposed approach, the tracked node transmits a signal and infrastructure nodes measure the Doppler shifts of the transmitted signal. We show that Mica2 motes can measure RF Doppler shifts with 0.2 Hz accuracy corresponding to a 0.14 m/s error in relative speed estimates using radio interferometric technique.

The tracking problem is modeled as a non-linear optimization problem and an extended Kalman filter is used to solve it accurately assuming Gaussian measurement errors. However, this approach may fail if the tracked node changes its speed or direction. We propose to update the Kalman filter state by performing constrained least-squares optimization when a maneuver is detected. The combined approach achieves almost a 50% accuracy improvement over the Kalman filter alone when the mobile node changes its direction and speed frequently. We describe our proof-of-concept implementation of the tracking service and evaluate its performance experimentally and in simulation.

Categories and Subject Descriptors: C.2.4 [Computer-Communications Networks]: Distributed Systems

Keywords: Sensor Networks, Doppler effect, Tracking

General Terms: Algorithms, Experimentation, Theory

Acknowledgments: ARO MURI grant W911NF-06-1-0076, NSF CNS-0347440 grant and a Vanderbilt Discovery grant have supported, in part, the research described in this paper. We would like to express our gratitude to Prabal Dutta, Gyorgy Balogh and Peter Volgyesi for their valuable contributions to this work.

SenSys'07, November 6–9, 2007, Sydney, Australia.

1 Introduction

Wireless sensor networks (WSNs) are recognized for their potential to connect the physical world with the virtual world of computers through integrated, low-power, resource-constrained devices [8, 16]. In fact, the demonstrated success of WSNs in detection, classification, monitoring and management of objects in the physical world [12, 17, 31, 4] is due to the large number of sensors that can be embedded in the environment close to the phenomena of interest over wide areas. The main strength of WSNs is achieved through distributed collaboration, thus overcoming the limitations of a single sensor. Therefore, the spatio-temporal coordination of sensors becomes important in many WSN applications.

In this paper, we consider the problem of keeping track of the accurate locations and velocities of mobile sensors continuously over time. Tracking of mobile objects has been an active area of research [2, 24, 6]. Generally, resource constraints of WSNs limit the algorithms that can be implemented as well as the amount of memory used and ranging data exchanged.

We argue that the most power-efficient way to localize mobile sensors is not to localize them at all when they are not moving. Sensors should be responsible for maintaining their state, including their location, and invoke the tracking service only if they detect that the location has changed. Cheap and power efficient accelerometers are well suited for this purpose. In our approach, the initial locations of the sensors are found at deployment time using one of the computationally more expensive methods such as in [9, 26, 27]. When a node detects that its location has changed, it notifies the tracking service and a number of stationary sensors deployed in the vicinity are assigned to participate in tracking. Afterwards, the location and velocity information are sent back to the node, thus keeping its state estimate accurate at all times.

Recently, a radio interferometric technique was proposed for the precise localization of stationary nodes [26, 20], as well as for tracking mobile nodes [19]. The approach utilizes interfering radio signals transmitted by two nodes simultaneously. The relative phase offset of the interference signal at two different receivers can be used to estimate distances among the four nodes involved.

We propose to utilize a similarly constructed interference signal, but instead of measuring its phase, we measure the Doppler shifts caused by moving nodes to estimate their *ve-*

locity and location simultaneously. Consequently, our approach allows for simpler and faster tracking:

1. Each of the receivers measures the Doppler shift only, whereas the relative phase measurements need to be carried out by pairs of receivers in [22], and
2. Doppler shifts are measured at a single carrier frequency, whereas up to 21 different frequencies are required to obtain accurate ranging data from phase measurements in [22].

The goal of our tracking algorithm is to estimate the location and the velocity of the tracked node from Doppler shift measurements. A Kalman Filter (KF) [18] can be used to track the mobile sensor under a constant-velocity assumption. However, it is a well known problem that KF fails to track sudden maneuvers of the tracked object [6]. To improve the tracking accuracy in the maneuvering case, we propose a simple maneuver detection and compensation algorithm and show that this improves the KF accuracy by as much as 50%. Specifically, the maneuver is detected based on a significant change of the heading or the speed of the tracked node. Upon detection, we combine the Extended Kalman Filter (EKF) with Constrained Non-linear Least Squares (CNLS) optimization as follows: EKF is used to obtain the initial state estimate that predicts the region where the tracked node is located. The CNLS optimization is then used to find a more accurate state estimate within the region of interest and the EKF is updated with the new state estimate. Since the CNLS optimization is sensitive to measurement errors, we only use it for a short time after the maneuver was detected. Otherwise, EKF is used alone as it works well under the constant-velocity assumption. We refer to the combined approach as the CNLS-EKF algorithm.

We evaluated the CNLS-EKF algorithm both experimentally and in simulation. The experimental data was collected in an outdoor area by testing two scenarios: (a) the tracked node was moving at a constant speed with infrequent direction changes, and (b) the node frequently changed both its speed and direction. The corresponding location accuracy achieved was 1.3 and 2.2 m, respectively, and the precision of the speed and heading estimates were $0.1-0.4$ m/s and $7-18^{\circ}$, respectively. We have also tested the CNLS-EKF algorithm using thousands of simulated measurements, studying how the tracking accuracy is affected by the number of participating infrastructure nodes, the maximum speed of the tracked node, and the tracking update rate.

The rest of the paper is organized as follows. We state the problem and discuss the proposed approach in Section 2. In Section 3, we describe the technique used to measure the Doppler shifts. Next, we present the description of the CNLS and EKF techniques and apply them to our tracking problem. We provide implementation details in Section 5 and evaluate our approach both in simulation and experimentally in Section 6.

2 Tracking Service Design

Precise localization of wireless nodes have many current and potential future applications. One of the driving application we envision is the precise tracking of assets in a warehouse. While passive RFID systems (and even barcode-based solutions) exist, the location information they provide is only the proximity to a reader at a time instant. We envision a system that can locate boxes and pallets in a warehouse within a meter in 3D at any given time.

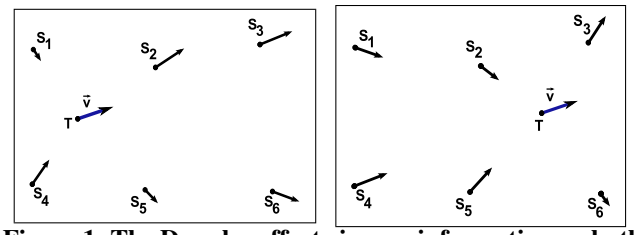

Figure 1. The Doppler effect gives us information on both the velocity and the location of a moving object.

Here is how the system could work. Simple and inexpensive tags are attached to the boxes in a warehouse. Each tag is in low-power mode until a simple accelerometer or mechanical movement sensor wakes it up. The tag broadcasts its ID and waits for a command to start ranging measurements which is repeated every few seconds. Once motion is not detected, the tag goes back to sleep after a timeout. The warehouse itself is instrumented with a network of infrastructure nodes. As they receive messages from tags, they schedule ranging measurements. The localization algorithm determines the tag location and stores it in a database. When an inquiry is made for a given tag, its last position (or even tracks of its past movement) is immediately available.

To summarize, we impose the following constraints on the design of our tracking service: (a) a number of infrastructure nodes are deployed at known locations in the area of interest, (b) tracked nodes cooperate with the infrastructure nodes to find their locations and velocity vectors, and (c) the hardware capabilities of both the infrastructure and tracked nodes are limited. Consequently, the design of our tracking service mandates relatively low sampling rates and algorithms with limited memory, computation, and communication requirements.

2.1 Approach

A well known phenomenon which is observed when objects move relative to each other is the Doppler effect. The Doppler effect law states that if an object transmits a signal and moves relative to an observer, the frequency of the observed signal will be Doppler shifted and the magnitude of the shift depends on the frequency of the signal and the velocity of the transmitter and observer relative to each other. We suggest the following approach: a moving node transmits a signal at a known frequency and the frequency of the Doppler shifted signal is measured by the stationary infrastructure nodes. The speed of the tracked node relative to all infrastructure nodes can be calculated and used to find both the velocity and the location of the node.

Figure 1 shows tracked node T which is moving in an area where six infrastructure nodes are deployed (S_i). The velocity vector $\overrightarrow{v}$ of T is shown at two different locations in the two figures. The magnitude of the Doppler shifts observed at node S_i depends on the relative speed of T and S_i which can be found by projecting the velocity $\overrightarrow{v}$ on the TS_i line.

We plot the projected vectors for all sensors S_i in both figures. The length of the projected vectors depends on both the velocity vector $\overrightarrow{v}$ and the location of T. The vector $\overrightarrow{v}$ is the same in both figures, yet, the corresponding projected vectors have different lengths.

Note that there is an important difference between the information that is obtained by measuring Doppler shifts and the well studied bearing-only tracking systems that use passive radar, sonar, or infrared sensors to determine the angle of arrival of the signal. In our case, we can only determine the length of the projection vectors, whereas the bearing of these vectors remain unknown. Therefore, it is not possible to use triangulation to find the target coordinates. However, we will show that by measuring sufficiently many relative speeds, both the location and the velocity vector of the tracked node can be found accurately.

Our approach follows the general structure of many of the existing localization and tracking algorithms [2, 27, 9]:

Coordination phase: infrastructure nodes are notified to participate in tracking of a node in a certain region. Both the timeframes and the local coordinate systems of the participating nodes usually need to be synchronized to enable the data fusion of spatially and temporally distributed measurements.

Measurement phase: ranging measurements that provide information on the location, bearing, and/or speed of the tracked object are collected. The low level data is stored locally or handed off to a different node, for example, a higher level data fusion node.

Tracking phase: non-linear optimization and filtering techniques are used to smooth out the measurement noise by combining the ranging data measured at multiple infrastructure nodes at subsequent time steps. The movement of the tracked node can be predicted and the infrastructure nodes participating in the tracking can be activated or deactivated accordingly.

Even though our algorithm follows this general structure, there is a number of design choices and challenges left to be solved when devising our approach.

2.1.1 Coordination Phase

We assume that the tracked node cooperates with the tracking system which makes its detection trivial. However, we still need to identify the infrastructure nodes that will participate in the tracking. This is achieved by the tracked node broadcasting a tracking-request. All infrastructure nodes that receive the request will participate in the tracking. We will later show that the effective range of our ranging method is more than the actual communication range, therefore, all participating nodes will acquire meaningful data.

We assume that the locations of the infrastructure nodes are pre-surveyed with sufficient accuracy, thus avoiding the need for localization. Since the infrastructure nodes are assumed to be stationary, this is a one-time task which can be done during the deployment of the tracking system. However, the tracked node and the participating infrastructure nodes need to time synchronize with relatively high accuracy. This is required by the ranging method that we use as well as to allow for the data fusion of the ranging data. It was shown in [21] that a single radio message can be used to accurately synchronize the transmitter and all recipients of that message. We use a similar approach. The tracking-request message broadcasted by the tracked node is used as the synchronization point, allowing us to achieve a few-microsecond accuracy.

2.1.2 Measurement Phase

The tracked node transmits a signal and multiple infrastructure nodes measure the Doppler shifts of this signal. The main challenge here is to select the type of signal and the measurement method, so that the Doppler shifts can be measured with sufficient accuracy using low-cost sensor network hardware. We describe our measurement method in Section 3 in more detail. The implementation on the Mica2 mote platform is discussed in Section 5.

2.1.3 Tracking Phase

An Extended Kalman Filter (EKF) is used to estimate the location and velocity of the tracked node from the Doppler shift measurements obtained in the measurement phase. When a maneuver is detected, we update the Kalman filter state by running Constrained Non-linear Least Squares (CNLS) optimization on the last set of collected measurements. More details on how both the CNLS and EKF techniques are applied to our tracking problem can be found in Section 4.

3 Measuring Doppler Shifts

Measuring the frequency of a given signal with sufficient accuracy becomes a challenge when resource constrained hardware with limited sampling rate needs to be used. A popular and efficient way to determine the frequency of the signals is frequency domain analysis. However, it has been shown in [10] that computing the frequency spectrum by FFT is prohibitively expensive given our typical platforms. In particular, it would take approximately 15 seconds to calculate a 512-point FFT using an 8 MHz processor typically available in many of the commercial sensor nodes.

Time domain analysis, on the other hand, requires the frequency of the original signal to be relatively small due to the sampling rate limitations of sensor network hardware. The magnitude of the Doppler shift observed at a given velocity is proportional to the frequency of the measured signal — the higher the frequency, the larger the observed frequency shift. Therefore, decreasing the frequency of the signal results in smaller Doppler shift which in turn requires improved measurement accuracy.

Sensor networks are well suited for two types of the transmitted signals: acoustic and radio, as both can be generated and detected by hardware with relatively low incremental cost. The typical frequency of acoustic buzzers is $1-5$ kHz and the corresponding Doppler shift is $3-15$ Hz per 1 m/s velocity. Due to the relatively low frequency of the acoustic signals, they can be analyzed directly utilizing the resource constrained WSN hardware.

Radio signals are favored over acoustic signals, because sensor nodes are equipped with the radio transceivers for wireless communication. Moreover, radio transmission is unobtrusive and less prone to the interference from the environment. The Doppler shift observed at the typical carrier ra-

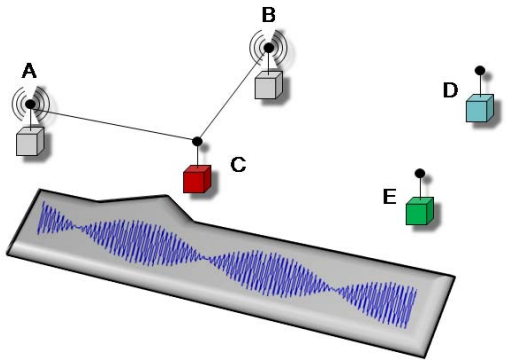

Figure 2. Two transmitters A and B transmit at the same time at two close frequencies. The interfere signal is observed by receivers C,D, and E.

dio frequencies (400 MHz − 2.4 GHz) is 1.3 − 8 Hz per 1 m/s velocity. The problem is that these radio frequencies are too high to be analyzed directly. Radio interferometry was proposed in [26] as a technique that allows to indirectly analyze high frequency radio signals with low-cost hardware. We show that this technique can be used to measure RF Doppler shifts with sufficient accuracy and consequently, utilize radio signals for tracking in this paper.

3.1 Radio Interferometry

The radio Interferometric Positioning System (RIPS) was proposed in [26] to derive location information by analyzing the phase of radio signals with low cost hardware. In this approach, two nodes transmit sine waves at different frequencies at the same time, so that the two signals interfere with each other. It can be shown that if f_a and f_b are the frequencies of the two transmitted sine waves, then the resulting interference signal has a frequency of $(f_a + f_b)/2$ and a low frequency envelope of $|f_a - f_b|$ (see [26]). Fig. 2 shows an example of the interference signal and its low frequency beats at node C. The theoretical model of the radio interference was developed in [26] to link the phase difference of the interference signal at two different receivers to the so called *q-range* quantity — a linear combination of distances between the two transmitters and two receivers. By measuring multiple of these q-ranges, it is possible to achieve high accuracy localization.

The main advantages of the RIPS approach are: (a) it requires no extra hardware because common radio transceivers can be utilized; (b) it utilizes resource constrained hardware to analyze the interfering radio signals because the frequency of the beats can be tuned to be low; and (c) it allows for large range (about twice the radio range) because it is the phase of the radio signals that is analyzed, rather than their amplitude.

We have recently shown that simultaneous tracking of multiple nodes is possible using the RIPS technique and that the tracking accuracy of mobile nodes can be significantly improved by measuring the Doppler shifts in the interference signal [22] as well. However, the Doppler shifts alone were not and could not be used for tracking in this approach as they did not carry enough information to calculate the locations of the tracked nodes. This was because the tracked nodes were assigned to participate in RIPS as receivers, to allow for simultaneous tracking of multiple nodes. Consequently, each tracked node could only determine one speed-related quantity per ranging measurement which is clearly not enough to find its location. In this paper, we assign the tracked node to be one of the transmitters and show that each infrastructure receiver can calculate the relative speed of the tracked node. Thus, a number of infrastructure nodes can collect sufficiently many measurements to determine the location of the tracked node from the Doppler shifts alone.

3.1.1 New contributions

We propose that the tracked node T is a transmitter (Fig. 3). Let T transmit an unmodulated sine wave at frequency f_t and let an infrastructure node A transmit a sine wave at frequency f_a, such that $f_t > f_a$. The two sine waves interfere with each other and create a signal with an envelope frequency of $f_t - f_a$. The interference signal is measured by a number of infrastructure nodes S_i. Since T moves relative to the infrastructure nodes, Doppler shifted frequencies will be observed. The signal transmitted at f_t will be Doppler shifted by Δf_t^i at each node S_i where the magnitude of Δf_t^i depends on the relative speed of T and S_i. Infrastructure nodes do not move and the signal transmitted by A is not Doppler shifted. Therefore, the measured envelope frequency f_i of the interference signal at node S_i is given by

$$f_i = f_t - f_a + \Delta f_t^i. \quad (1)$$

Eqn. 1 allows us to calculate the Doppler shift measured at node S_i and consequently, the relative speed of the tracked node. We will later show that by measuring sufficiently many frequencies f_i, we can estimate the location and velocity of the tracked node.

Our approach differs from the RIPS algorithm in that we do not measure the phase of the interference signal. A disadvantage is that we loose the range information which can be deduced from the relative phase of two receivers. However, simpler and faster tracking algorithm can be implemented: (a) measuring the frequency of the interference signal allows for simpler and faster time-domain analysis of the interference signal, (b) a single receiver can compute the Doppler shift in our case, whereas a pair of receivers was required to measure the range-yielding relative phase differences in RIPS, thus allowing for a simpler data fusion algorithm, and (c) it is sufficient to measure the Doppler shift at a single carrier frequency, whereas the RIPS approach utilized multiple carrier frequencies. In fact, up to 50 different carrier frequencies were used in large scale deployments of RIPS [20] which significantly increased the measurement time.

One of the major advantages of our approach over RIPS is that each tracked node is required to transmit only at a single radio frequency. Consequently, each tracked node can be assigned a unique frequency for the ranging measurements and thus multiple nodes can be tracked in parallel in the same area without interfering with each other.

3.2 Doppler Shifts of the Interference Signal

We express the Doppler shifted frequencies measured by the infrastructure nodes as a function of the location and ve-

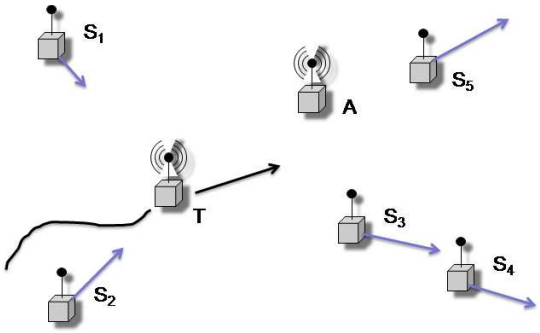

Figure 3. The mobile node T is being tracked by five infrastructure nodes S_i. One other node, A co-transmits with T. The receivers S_i calculate the relative speed of T (blue arrows) from the measured the Doppler shifts.

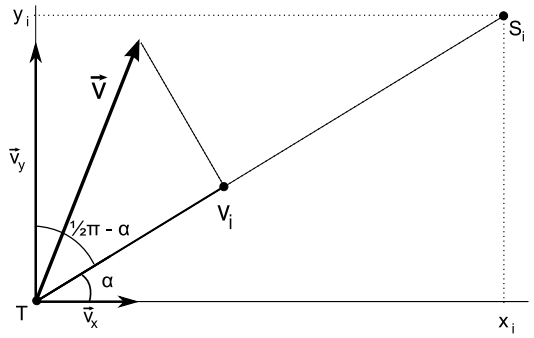

Figure 4. Tracked node T having velocity $\overrightarrow{v}$ transmits a signal. Sensor S_i measures the Doppler shift of the signal which depends on v_i, the relative speed of T and S_i.

locity of the tracked node in this section. Suppose that $\overrightarrow{v}$ is the velocity of the tracked node T and $\overrightarrow{u_i} = \overrightarrow{S_iT}/\|S_iT\|$ is the unit length vector pointing from sensor S_i to T. The relative speed of S_i and T can be defined as the following dot product

$$v_i = \overrightarrow{u_i} \cdot \overrightarrow{v}. \tag{2}$$

Note that v_i is a scalar value with positive sign if $\overrightarrow{v}$ points away from S_i and negative sign otherwise.

The Doppler equation states that if f is the frequency of the transmitted radio signal, c is the speed of light, and $v \ll c$ is the speed of the source with respect to the observer (with negative sign of v if the source was going towards the observer), then the observed frequency is $f' = (1 - v/c)f$. Therefore,

$$\Delta f = f' - f = -vf/c. \tag{3}$$

We apply the Doppler equation (3) to the interferometric equation (1). As mentioned before, only the Doppler shift in the frequency f_t will be observed (node A is stationary). Using $\widehat{f} = f_t - f_a$ and $\lambda_t = c/f_t$, the node S_i observes the interference signal with frequency f_i:

$$f_i \quad = \quad \widehat{f} - v_i/\lambda_t. \tag{4}$$

The frequency f_i can be measured at node S_i. Consequently, Eqn. (4) allows us to compute the relative speed of the tracked node T and the node S_i, if the difference of the two transmitted frequencies $\widehat{f}$ is known. Note that estimating the frequency difference $\widehat{f}$ with sufficient accuracy becomes a problem when using low-cost radio transceivers (see Section 5 for more details). Therefore, we have to treat $\widehat{f}$ as an unknown parameter in our tracking algorithm.

4 Tracking

Our tracking algorithm utilizes RF Doppler shifts measured by multiple infrastructure nodes to calculate the location and the velocity of a tracked node. We model the tracking problem as a non-linear optimization problem and use a combination of standard techniques to solve it.

4.1 Tracking as Optimization Problem

The parameters that need to be estimated are the location (x,y) and the velocity vector $\overrightarrow{v} = (v_x, v_y)$ of the tracked node. Also, one of the limitations of the low-cost radio chip that we use is that the actually transmitted frequency differs from the nominal frequency by up to a few kHz. This prevents us from determining the exact value of the frequencies f_t and f_a, hence their difference $\widehat{f}$ becomes one of our parameters. Consequently, we define the parameter vector $\mathbf{x}$ to be estimated as

$$\mathbf{x} = (x, y, v_x, v_y, \widehat{f})^{\mathsf{T}}.$$

Assuming that n infrastructure nodes measure the Doppler shifted radio signal, we have n frequency observations (f_i). We define the observation vector $\mathbf{c}$ as

$$\mathbf{c} = (f_1, f_2, \ldots, f_n)^{\mathsf{T}}.$$

The parameter vector $\mathbf{x}$ and the observation vector $\mathbf{c}$ are related to each other. We formalize this relation through a function $H : \mathcal{R}^5 \rightarrow \mathcal{R}^n$, such that

$$\mathbf{c} = H(\mathbf{x}).$$

The function H is a vector function consisting of n functions $H_i : \mathcal{R}^5 \rightarrow \mathcal{R}$, each of them calculating the Doppler shifted interference frequency f_i measured at an infrastructure node S_i. From Eqn. 4, $H_i(\mathbf{x})$ is defined as

$$H_i(\mathbf{x}) = \widehat{f} - v_i/\lambda_t.$$

The relative speed v_i of the nodes T and S_i can be calculated from the locations of the two nodes and the velocity $\overrightarrow{v}$ of the tracked node T (see Fig. 4). We use the distributive property of the dot product, and identities $\overrightarrow{v} = \overrightarrow{v_x} + \overrightarrow{v_y}$ and $|\overrightarrow{u_i}| = 1$ (the length of a unit vector is 1) to express v_i:

$$\begin{aligned} v_i \quad &= \quad \overrightarrow{u_i} \cdot \overrightarrow{v} = \overrightarrow{u_i} \cdot \overrightarrow{v_x} + \overrightarrow{u_i} \cdot \overrightarrow{v_y} \\ &= \quad |\overrightarrow{v_x}| \cos\alpha + |\overrightarrow{v_y}| \sin\alpha \end{aligned}$$

Finally, $\sin\alpha$ and $\cos\alpha$ can be calculated using the coordinates (x,y) and (x_i,y_i) of the nodes T and S_i, respectively. This allows us to calculate the expected measurements $f_i = H_i(\mathbf{x})$ from the parameters $\mathbf{x}$ and the known quantities λ_t, x_i, y_i.

$$H_i(\mathbf{x}) \quad = \quad \widehat{f} - \frac{1}{\lambda_t} \frac{v_x(x_i - x) + v_y(y_i - y)}{\sqrt{(x_i - x)^2 + (y_i - y)^2}} \tag{5}$$

Due to measurement errors, there may exist no $\mathbf{x}$ such that $H(\mathbf{x}) = \mathbf{c}$. Instead, we estimate the parameters by finding $\mathbf{x} \in \mathcal{R}^5$ such that $\|H(\mathbf{x}) - \mathbf{c}\|$ is minimized. Note that components of our objective function H are non-linear functions, requiring the use of non-linear optimization methods.

4.2 Non-linear Least Squares

Non-linear optimization techniques typically start with an initial approximation of the parameter vector $\mathbf{x_0}$ and iteratively update this parameter vector until it converges to a local minimum of an objective function. We use the Gauss-Newton method ([25]) which is based on a linear approximation of the objective function in the neighborhood of $\mathbf{x}$. For example, our objective function $H(\mathbf{x}) - \mathbf{c}$ is linearized by Taylor expansion as

$$H(\mathbf{x}+\Delta) - \mathbf{c} \simeq l(\Delta) = H(\mathbf{x}) - \mathbf{c} + J(\mathbf{x})\Delta,$$

where $J \in \mathcal{R}^{n\times 5}$ is the Jacobian of $H(\mathbf{x}) - \mathbf{c}$.

The detailed description of the Gauss-Newton algorithm follows:

Assuming the i-th parameter vector $\mathbf{x_i}$ is given,

(1) calculate Jacobian $J_i = J(\mathbf{x_i})$ and linearize the objective function around $\mathbf{x_i}$ (denoted by $l_i(\Delta)$),

(2) calculate a local minimizer Δ_i of the function $l_i(\Delta)$, and

(3) set $\mathbf{x_{i+1}} = \mathbf{x_i} + \alpha\Delta_i$, where α is the step length influencing the convergence of the method. Stop if Δ_i is small.

An additional problem is that $H : \mathcal{R}^5 \rightarrow \mathcal{R}^n$ is a vector function. Non-linear Least Squares (NLS) techniques define a new objective function $\mathcal{H} : \mathcal{R}^5 \rightarrow \mathcal{R}$ as

$$\begin{aligned} \mathcal{H}(\mathbf{x}) &= \frac{1}{2}\sum_{i=1}^{n}(H_i(\mathbf{x}) - c_i)^2 \\ &= \frac{1}{2}(H(\mathbf{x}) - \mathbf{c})^{\mathsf{T}}(H(\mathbf{x}) - \mathbf{c}). \end{aligned} \quad (6)$$

$\mathcal{H}$ is linearized using Taylor expansion as follows

$$\mathcal{H}(\mathbf{x}+\Delta) \simeq L(\Delta) = \frac{1}{2}l(\Delta)^{\mathsf{T}}l(\Delta).$$

Consequently, the formula for the gradient of L is

$$L'(\Delta) = J(\mathbf{x})^{\mathsf{T}}(H(\mathbf{x}) - \mathbf{c}) + J(\mathbf{x})^{\mathsf{T}}J(\mathbf{x})\Delta,$$

which after letting $L'(\Delta) = 0$ gives us the solution for the local minimizer of $\mathcal{H}$ around $\mathbf{x}$

$$\Delta_{\min} = (J(\mathbf{x})^{\mathsf{T}}J(\mathbf{x}))^{-1}J(\mathbf{x})^{\mathsf{T}}(\mathbf{c} - H(\mathbf{x})). \quad (7)$$

More detailed derivation of this equation can be found in [25].

4.2.1 *Constrained Optimization*

In tracking, we are often able to constrain the area where the tracked node is located. Therefore, applying constrained non-linear least squares (CNLS) may yield more accurate results. One way to solve the constrained optimization is to modify the objective function by adding a barrier function to it, introducing zero penalty inside the region of interest and positive penalty outside of it.

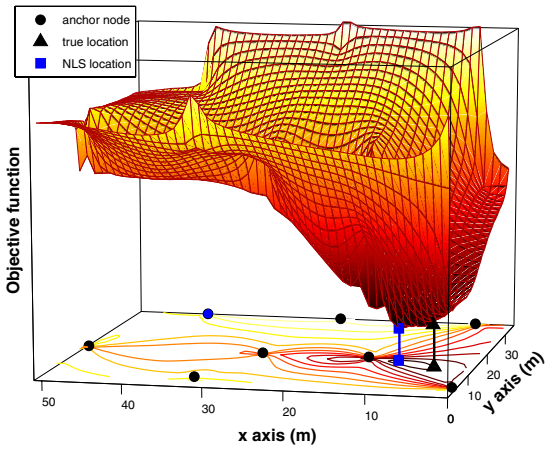

Figure 5. The objective function $\mathcal{H}$ minimizes the least-square error in the observation vector. Due to measurement errors, the global minimum of $\mathcal{H}$ is not at the true location of the tracked node.

We want to constrain the location of the tracked node to a disk centered at (x_0, y_0), with radius r. A logarithmic barrier function $\mathbf{b}(\mathbf{x})$ can be defined as

$$\mathbf{b}(\mathbf{x}) = -\log(r - \sqrt{(x-x_0)^2 + (y-y_0)^2}),$$

where $\mathbf{x} = (x, y, v_x, v_y, \widehat{f})$ is the parameter vector. Note that as $\sqrt{(x-x_0)^2 + (y-y_0)^2}$ approaches the radius r, the logarithm goes to $-\infty$ and the penalty function goes to ∞.

The CNLS algorithm works as the NLS algorithm, except it optimizes the function $\mathcal{H}(\mathbf{x}) + \mathbf{b}(\mathbf{x})$. The derivatives used in the NLS algorithm need to be adjusted by adding the term $B(\mathbf{x}) = \dfrac{\partial \mathbf{b}(\mathbf{x})}{\partial \mathbf{x}}$ to $L'(\Delta)$. Consequently, Eqn. (7) becomes

$$\Delta_{\min} = (J(\mathbf{x})^{\mathsf{T}}J(\mathbf{x}))^{-1}[J(\mathbf{x})^{\mathsf{T}}(\mathbf{c} - H(\mathbf{x})) + B(\mathbf{x})]. \quad (8)$$

4.2.2 *Problems with NLS optimization*

Non-linear least squares optimization may fail depending on the starting point $\mathbf{x_0}$ and the measurement errors that corrupt the observation vector $\mathbf{c}$. This is because the solution will converge to a local minimum of the objective function, or because the observations are insufficient to determine the parameter vector accurately.

We confirmed that our objective function $\mathcal{H}$ is susceptible to these problems experimentally. We placed eight infrastructure nodes in a 30×50 m area on our campus and measured the Doppler shifted frequency of the transmitted signal. In Fig. 5, we show the locations of the infrastructure nodes as black dots and the location of the transmitter as a triangle. Recall that $\mathbf{x}$ consists of five parameters x, y, v_x, v_y, and $\widehat{f}$. The function plotted in the figure is obtained by finding the minimum value of $\mathcal{H}(\mathbf{x})$ for the fixed coordinates (x, y) (i.e., finding the best fit for the three remaining parameters). Figure 6 shows the best-fit velocities found at a given location.

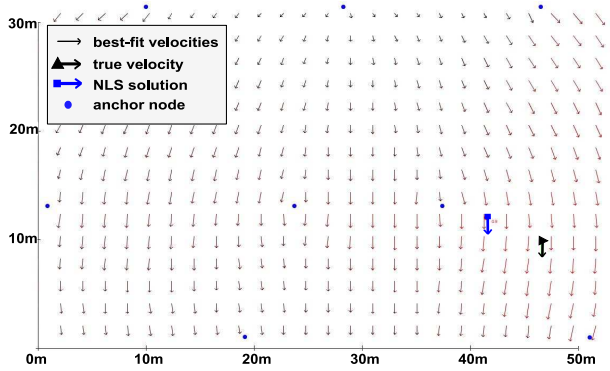

Figure 6. The best-fit velocities of the tracked node that minimize the function $\mathcal{H}$ in Fig. 5 are shown.

We make two observations: (a) the function $\mathcal{H}$ indeed contains multiple local minima close to the location of the tracked node, thus the constrained NLS algorithm is required, and (b) the global minimum of $\mathcal{H}$ (square) is 5.6 m away from the true location of the transmitter (triangle), therefore, all optimization techniques will introduce a large localization error in this particular case. Since the optimization methods fail to find the correct location of the tracked node in certain cases, we need to find alternate solutions to our tracking problem.

On the positive side, the objective function is relatively smooth and converges fast to the general area where the tracked node is located. Also, the estimated velocity of the tracked node, as shown in Fig. 6, is accurate in a relatively large area around the true location of the tracked node. This allows us to use the velocities calculated with the CNLS algorithm with higher confidence than the locations.

4.3 Extended Kalman Filter

Noise corrupted observations may prevent us from solving the tracking problem with sufficient accuracy, as shown in the previous section. Therefore, we resort to state estimation techniques which model the dynamics of the tracked node, estimate the state of the node based on its dynamics, and update the state based on the new observations. The Kalman filter (KF) is a widely used method for the state estimation of dynamic systems using a set of noisy measurements. The current state $\mathbf{x_k}$ of the system is calculated recursively from the previous state $\mathbf{x_{k-1}}$ and a new observation vector $\mathbf{c_k}$. An error covariance matrix $\mathbf{P}$, a measure of the accuracy of the estimated state $\mathbf{x}$, is calculated along with $\mathbf{x}$.

An important step in designing a KF is modeling the dynamics of the observed system. The KF framework requires that the system state $\mathbf{x_k}$ evolves according to

$$\mathbf{x_k} = \mathbf{F}\mathbf{x_{k-1}} + \mathbf{w_k},$$

where $\mathbf{x_{k-1}}$ is the previous state, $\mathbf{F}$ models the system dynamics, and $\mathbf{w_k}$ is the process noise with covariance $\mathbf{Q}$.

The KF framework also defines the model of the measurement process which is non-linear with respect to $\mathbf{x}$ in our case. An Extended Kalman Filter (EKF) can be applied which essentially linearizes the measurement function by calculating its Jacobian around the current state estimate. The observation $\mathbf{c_k}$ of the current state is modeled as

$$\mathbf{c_k} = H(\mathbf{x_k}) + \mathbf{v_k},$$

where H is the non-linear measurement function and $\mathbf{v_k}$ is normally distributed measurement noise with covariance $\mathbf{R}$.

$\mathbf{x_k}$ and $\mathbf{P_k}$ constitute the state $\mathcal{S}$ of the EKF. This state is updated in two phases: (a) a time prediction phase during which the current state is estimated based on the previous state, and (b) a measurement update phase during which the new measurements are used to refine the predicted state.

KF prediction phase. Given the previous state of the KF $(\mathbf{x_{k-1}}, \mathbf{P_{k-1}})$, the new state $(\mathbf{x_k^-}, \mathbf{P_k^-})$ is

$$\begin{aligned} \mathbf{x_k^-} &= \mathbf{F}\mathbf{x_{k-1}} \\ \mathbf{P_k^-} &= \mathbf{F}\mathbf{P_{k-1}}\mathbf{F}^\mathsf{T} + \mathbf{Q}. \end{aligned} \tag{9}$$

EKF update phase. The predicted state $(\mathbf{x_k^-}, \mathbf{P_k^-})$ is updated with $\mathbf{c_k}$ and the new state $(\mathbf{x_k}, \mathbf{P_k})$ is

$$\begin{aligned} \mathbf{K_k} &= \mathbf{P_k^-}\mathbf{J_k}^\mathsf{T}(\mathbf{J_k}\mathbf{P_k^-}\mathbf{J_k}^\mathsf{T} + \mathbf{R})^{-1} \\ \mathbf{x_k} &= \mathbf{x_k^-} + \mathbf{K_k}(\mathbf{c_k} - H(\mathbf{x_k^-})) \\ \mathbf{P_k} &= (\mathbf{I} - \mathbf{K_k}\mathbf{J_k})\mathbf{P_k^-}, \end{aligned} \tag{10}$$

where $\mathbf{R}$ is the measurement noise, $\mathbf{K_k}$ is Kalman gain, and $\mathbf{J_k}$ is the Jacobian matrix of H in $\mathbf{x_{k-1}}$.

4.3.1 Model of our system

The system state $\mathbf{x} = (x, y, v_x, v_y, \widehat{f})$ and the measurement vector $\mathbf{c} = (f_1, \ldots, f_n)$ are equivalent to the parameter and observation vectors defined in Section 4.1, respectively. Both, the process noise $\mathbf{w}$ and the measurement noise $\mathbf{v}$ are assumed to have "white noise" properties with zero mean and their covariant matrices will be determined later experimentally.

The state transition matrix $\mathbf{F}$ of our linear state space is

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & t & 0 & 0 \\ 0 & 1 & 0 & t & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

where t is the time elapsed from the previous state. The non-linear observation function H was defined in Eqn. (5).

4.3.2 Problems with the Kalman filter

We applied the EKF to our tracking problem and found that the filter tracks the mobile nodes accurately, mitigating the effects of the measurement noise. However, this only works well if the tracked node moves at a constant speed and does not change its direction. Situations, when the tracked node changes its direction significantly, for example by 180°, are potentially the most severe. We illustrate this problem in Fig.7. We deployed a number of infrastructure nodes (black dots) and moved the tracked node at a constant speed. After some time, we changed both the direction and the speed of the tracked node and observed how the EKF handles this situation (dotted line). The track that the node followed consisted of two $\sim$ 30 m segments. As we can see, the filter requires six measurement rounds to converge back to the true location, introducing significant errors in the location estimates in this transitional period.

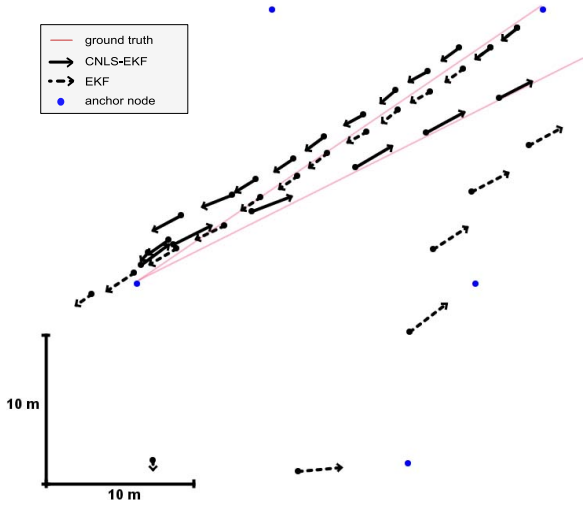

Figure 7. The EKF fails if the tracked node makes sudden maneuvers. The CNLS-EKF outperforms the EKF in the maneuvering case, while keeping good performance in the non-maneuvering case.

The divergence of the EKF in tracking maneuvering nodes is a well known problem. A simple solution is to increase the process noise covariance $\mathbf{Q}$, assigning more weight to the measurements than to the prediction model. This, however, degrades the overall tracking accuracy as the effects of the measurement noise become more significant. Other techniques propose to simultaneously use multiple Kalman filters which are set up with different models of the tracked node dynamics. Also, the EKF update algorithm can be executed iteratively, to better approximate the error function locally. The disadvantage of these techniques is their higher computational complexity, as the KF prediction and update need to be executed multiple times per observation.

4.3.3 Solving the maneuvering case

We propose to combine the EKF and the CNLS techniques in a way that significantly improves the tracking accuracy in the maneuvering case, while maintaining the good performance of the EKF in the non-maneuvering case. The combined algorithm is referred to as the CNLS-EKF algorithm. The main motivation for choosing CNLS is its fast convergence, given a good initial estimate of the parameters.

The CNLS-EKF algorithm proceeds in three steps:

1. Given a new observation vector $\mathbf{c}$, calculate the new EKF state $\mathcal{S} = (\mathbf{x}, \mathbf{P})$ using Eqns. (9) and (10),
2. If a maneuver is detected, find a new system state $\mathbf{x}^*$ by running the CNLS optimization initiated at $\mathbf{x}$, and
3. Use the new system state $\mathbf{x}^*$ to update the EKF state $\mathcal{S}$.

Maneuver detection algorithm:
In our experience, maneuvers can be detected reliably when the direction and the speed of the tracked node change significantly from their last estimates. This is because the velocity estimates are relatively robust to both measurement errors and the errors in predicted location (see Fig. 6).

CNLS-based EKF update:
Let $\mathbf{x}^*$ be the solution of CNLS. The EKF state $\mathcal{S} = (\mathbf{x}, \mathbf{P})$ is updated by running the linear KF update algorithm using $\mathbf{x}^*$ as the observation vector and the identity matrix $\mathbf{I_5}$ as the measurement matrix. In particular, the Kalman gain $\mathbf{K'_k}$ and the updated state $\mathcal{S}' = (\mathbf{x}', \mathbf{P}')$ are

$$\begin{aligned} \mathbf{K'_k} &= \mathbf{P}(\mathbf{P}+\mathbf{R}^*)^{-1} \\ \mathcal{S}' &= (\,\mathbf{x}+\mathbf{K'_k}(\mathbf{x}^*-\mathbf{x}), (\mathbf{I}-\mathbf{K'_k})\mathbf{P}\,). \end{aligned}$$

The covariance matrix $\mathbf{R}^*$ determines how much the CNLS solution $\mathbf{x}^*$ influences the state $\mathcal{S}'$. Since the CNLS optimization was shown to be sensitive to measurement errors, we need to limit its influence in the non-maneuvering case. Therefore, we define the covariance matrix $\mathbf{R}^*$ with exponentially increasing values over time. The more time has passed since the maneuver, the smaller the influence of $\mathbf{x}^*$ on the state $\mathcal{S}'$. If the time elapsed from the last detected maneuver is Δt seconds, we define $\mathbf{R}^*$ as

$$\mathbf{R}^* = \rho \begin{bmatrix} 5^{\Delta t} & 0 & 0 & 0 & 0 \\ 0 & 5^{\Delta t} & 0 & 0 & 0 \\ 0 & 0 & 2^{\Delta t} & 0 & 0 \\ 0 & 0 & 0 & 2^{\Delta t} & 0 \\ 0 & 0 & 0 & 0 & 2^{\Delta t} \end{bmatrix}.$$

The base of the exponential function is higher for the location coordinates than the velocity coordinates because the velocity estimates are less prone to measurement errors and the initial state choice (see Fig. 6). We keep the error of the interference frequency $\widehat{f}$ small, because it does not change significantly between consecutive measurements. The scaling factor ρ allows us to fine-tune the weight of the CNLS solution.

The improvement of the CNLS-EKF algorithm over regular EKF can be seen in Fig. 7. The tracking accuracy can improve by as much as 50% (see Section 6).

5 Implementation

We have implemented the proposed tracking system using the TinyOS operating system [23]. Our hardware platform is the popular Mica2 mote [15] which is a low-power, battery-operated wireless platform with an 8 MHz CPU, 4 kB memory and the CC1000 Chipcon radio transceiver [5]. The most important criterium for our platform choice is the functionality provided by the CC1000 radio chip (allowing implementation of the radio interferometric technique). During the actual field tests, we have used ExScal nodes [7] which are Mica2 compatible motes enclosed in a weather-proof packaging. In the current proof-of-concept implementation, the CNLS-EKF algorithm runs on a PC-class device, utilizing the Doppler shifts measured by the sensor nodes.

5.1 Creating the Interference Signal

The interference signal is created when two transmitters transmit simultaneously at different frequencies. If the two frequencies are almost the same, the resulting signal has a low frequency envelope which can be observed at the Radio Signal Strength Indicator (RSSI) pin of the radio chip. Since the frequency of the envelope equals the difference of the two transmitted frequencies, we need to set the frequencies

precisely in order to control the frequency of the observed interference signal. In fact, we need to keep the interference frequency in a relatively narrow range ($300-400$ Hz) to be able to analyze the signal using our computationally constrained hardware.

Note that the low-cost radio transceiver that we use does not allow for setting the transmission frequency accurately. We have observed ten parts-per-million (ppm) clock oscillator errors corresponding to ± 4 kHz frequency errors at the 400 MHz operating frequency of our radios. However, we need to know the actual radio frequencies f_t and f_a of the two transmitting nodes accurately as the measured Doppler shift is calculated from their difference $f_t - f_a$ and the observed interference frequency f_i, according to Eqn. (1). For this reason, we had to treat $\widehat{f} = f_t - f_a$ as one of the unknown parameters in our algorithm. Note that the accurate value of the wavelength $\lambda_t = c/f_t$ is also required in our equations to calculate the relative velocities from the Doppler frequency shifts (Eqn. (4)). However, it is easily seen that 4 kHz error at 400 MHz corresponds to a small wavelength error ($\sim 10^{-3}$cm) and using the approximate wavelength is sufficient.

Calibrating two nodes to transmit at the same frequency is an easier problem than calibrating both nodes to transmit at accurate absolute frequencies. For example, radio interference can be used to calibrate two transmitters as follows. One node transmits at its un-calibrated (thus arbitrarily shifted) radio frequency, while the second transmitter sweeps a relatively large frequency band around this frequency. A close-by receiver observes the interference of the two waves and can determine the time when the two nodes transmit at the same frequency which allows the second transmitter to calibrate its radio. This technique requires that the radio frequency of the transceiver can be changed in relatively small steps. The CC1000 radio chip allows to tune the transmitted frequency in 65 Hz steps which is sufficient (see [19] for more details).

5.2 Measuring Doppler Shifts

The radio signal that we analyze is sampled at 8.862 kHz at the RSSI pin of the CC1000 radio chip. The RSSI circuit applies a low pass filter to the incoming signal thus removing high frequency components from it. Therefore, only the beat frequency will be visible in the RSSI signal.

The calibration algorithm described in the previous section will maintain the beat frequency in a pre-defined operating range, such as $300-400$ Hz. We have implemented a simple time-domain algorithm that computes the average period of the beat signal in a fixed time window. We apply a moving average filter to smooth the incoming signal and, consequently, find all peaks in the filtered signal. A period is defined as the number of samples between any two consecutive peaks. Since we know the expected period (i.e., 22–30 samples given the 8.9 kHz sampling and the 300–400 Hz expected beat frequency), we do additional filtering by removing the periods that are outside of the expected interval. Consequently, the beat frequency is computed as the reciprocal of the average period p. This algorithm runs online and does not require any post-processing.

The accuracy of the frequency measurement will improve if we increase the fixed window in which we observe the signal. Alternatively, one can perform the frequency measurement multiple times for smaller windows and average these results to improve the precision. Thus, the accuracy of our algorithm can be increased at the price of a longer measurement time. On the other hand, the measurement time needs to be relatively short because the Doppler shift is changing as the tracked node moves. Based on these experiments, we have chosen to have 450 samples in our observation window and repeated the measurement 6 times. For these parameters, the overall measurement time is 0.4 sec and the standard deviation of the observed measurements is 0.21 Hz. We have also studied how the interference frequency changes over time, observing 1 Hz standard deviation of the transmitted frequency in consecutive measurements. Both the variance of the measurement process and the variance of the interference frequency are important design parameters for the EKF algorithm.

6 Results

To evaluate our tracking algorithm we ran two experiments outdoors and calculated the accuracy of the location and velocity estimates of the tracked node. We further examined how the number of participating infrastructure nodes, the maximum speed of the tracked node and the tracking update rate influenced the tracking accuracy in a number of randomly generated simulated scenarios.

6.1 Experiments

ExScal motes served both as stationary infrastructure nodes and the mobile tracked node. An online version of our tracking algorithm was running while a person was walking or running with the tracked node.

6.1.1 Setup description

We have utilized eight infrastructure nodes that measured the Doppler effect and deployed them in a 50×30 m area (see Fig. 8). Since two nodes are required to transmit in our approach, one more infrastructure node was used to co-transmit with the tracked node. We measured the ground truth locations of the infrastructure nodes with an estimated error of 0.5 m. The resulting network had two-hop diameter for the duration of the experiment.

It is hard to estimate the ground truth for the moving tracked node, both spatially and temporally. To simplify this task, we have limited the node's track to a series of straight line segments, connected at their endpoints (gray lines in Fig. 9). Moreover, the person carrying the node was walking or running at an approximately constant speed for each of the segments and varied the speed somewhat for different segments. The task of finding the ground truth of the whole track is then reduced to finding the ground truth for each segment.

Estimating the location and speed in a given line segment was accomplished by recording the times when the tracked node passed the endpoints of the given segment. The speed of the tracked node is calculated as the line segment length over the time it took to cover the segment. The location of the tracked node can be found by interpolating the segment

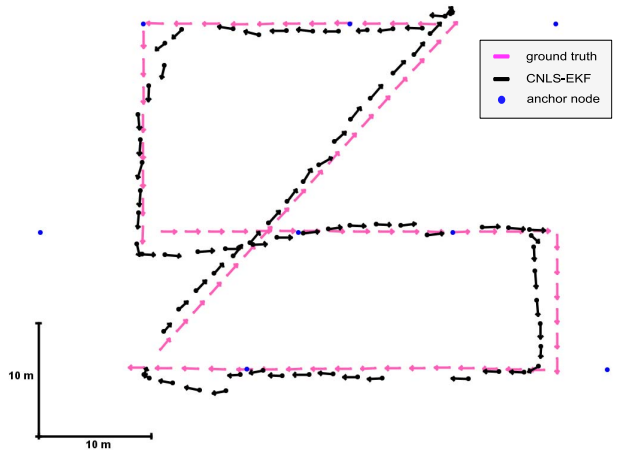

Figure 8. Experimental evaluation of the CNLS-EKF algorithm. Eight anchor nodes are shown as black dots. The location and the velocity of the tracked node are shown as a black dot and an arrow, respectively. The ground truth is shown in gray.

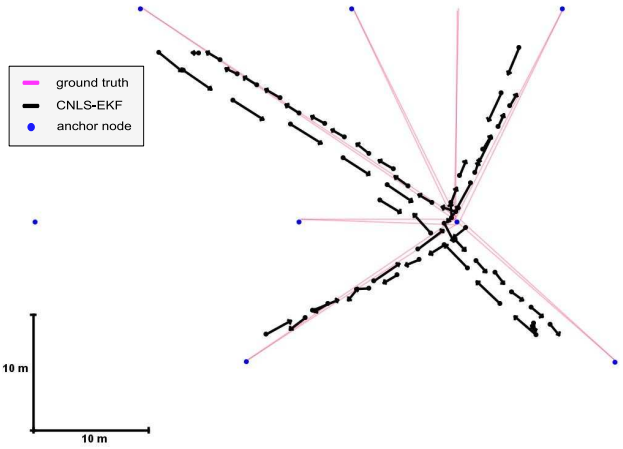

Figure 9. We show the worst-case situation for the Kalman filter: in a star-like topology, the tracked node moved slowly away from the center and then fast towards the center of the star. Results for only four out of seven tracks are shown for clarity.

line, using the measurement time as the interpolation coefficient. The calculated ground truth is shown by gray arrows in Fig. 8.

We believe that this process resulted in errors less than 1 m, 0.2 m/s, and 5° in the location, speed, and heading estimates, respectively.

6.1.2 Measurements

The update interval that we achieved in our experiments was 1.7 sec, coming from 0.3 sec coordination phase, 0.4 sec measurement time, and 1 sec time to route the measurements from the infrastructure nodes to a PC. The most computationally intensive is the 0.4 sec measurement phase, during which the receivers analyze the RSSI signal sampled at ~ 9 kHz, utilizing almost 100% of their CPU. However, apart from the measurement phase, the tracking service does not require significant system resources. This allows other WSN applications to run in parallel on the same sensor node. Occasionally, measurements from some of the nodes failed to reach the PC due to the routing collisions. We have considered the measurement round invalid, if fewer than 50% of the measurements were received. The CNLS-EKF algorithm was not invoked for invalid measurements.

We ran experiments that evaluate our algorithm in two cases: (a) the tracked object moves along straight lines for substantial amounts of time at a constant speed, and (b) the tracked object changes its speed and direction abruptly and significantly after relatively short periods of time. The Kalman filter assumption of constant speed is violated in case (b), resulting in degraded performance.

Experiment 1: The deployment setup is shown in Fig 8. The ground truth is shown in gray and the tracking algorithm results are indicated by the black arrows. The tracked node was moving at the mean speed of 1.3 m/s which was varying at most 0.2 m/s. We ran the CNLS-EKF algorithm on this data, setting the process noise covariance matrix Q with 0.5, the measurement noise covariance matrix R with 0.2, and the radius r of the barrier function $\mathbf{b}(\mathbf{x})$ with 3 m. The mean location, speed, and heading errors can be found in Table 1. We also ran EKF alone on the same data set and found that CNLS-EKF achieves only a modest accuracy improvement in this case.

Mean Errors:	**Location**	**Speed**	**Heading**
EKF algorithm	1.5 m	0.14 m/s	7.2°
CNLS-EKF algorithm	1.3 m	0.13 m/s	6.9°
Improvement over EKF	10%	1.7%	4.4%

Table 1. The mean errors were calculated based on 74 measurements collected during Experiment 1. The improvement over the EKF was modest.

Experiment 2: We evaluated the scenario which violated the assumption of the linear dynamics of the tracked node, based on which the Kalman filter was modeled. Both the speed and the heading of the tracked node was changed frequently. We selected a central point in our deployment area and designed the path to be followed by the tracked node as a star graph (see Fig 9). A person carrying the tracked node was walking (1.2 m/s) when moving away from the center. Upon reaching the outside endpoint, the person started running (up to 3 m/s) in the opposite direction, towards the center. We have measured 94 data points in this case and show the results in Table 2. The improvement over EKF with no maneuver correction was more significant this time. However, the speed and heading errors of EKF were similar to those of CNLS-EKF.

6.1.3 Discussion

Both EKF and CNLS-EKF algorithms perform well, if the dynamics of the tracked node is consistent with the EKF model. If the tracked node maneuvers relatively infrequently, the EKF is able to converge to the true location of the tracked node relatively fast after the maneuver. Consequently, the small additional error of the filter is averaged out for the whole track.

Mean Errors:	Location	Speed	Heading
EKF algorithm	4.3 m	0.42 m/s	17.7°
CNLS-EKF algorithm	2.2 m	0.35 m/s	17.5°
Improvement over EKF	48.7%	16.3%	0.4%

Table 2. We collected 94 measurements in Experiment 2. The improvement over the EKF was more significant, especially in location estimation.

However, if the tracked node changes its velocity significantly and the maneuvers are frequent, the EKF takes a long time to converge, or diverges completely. Consequently, the location error grows significantly. The constrained optimization is able to rapidly correct the Kalman filter state after a maneuver which results in faster convergence of the CNLS-EKF filter and in a better overall location accuracy. Notice that the velocity estimates have approximately the same errors for both EKF and CNLS-EKF. Intuitively, this is because our measurement model is based on estimating the relative velocities of the tracked node which gives us more information on its velocity than its location. Therefore, even if we optimize our objective function at a wrong location, the velocity estimates are still relatively accurate (see Fig. 6).

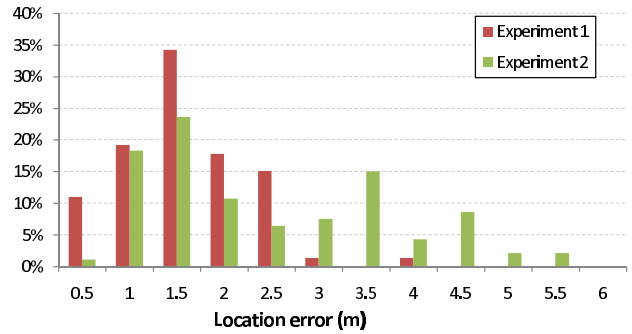

Figure 10. Distribution of the location estimation errors of the CNLS-EKF algorithm for both experiments.

The distributions of the location estimate errors of both experiments are shown in Fig. 10. The errors in the first experiment are approximately normally distributed around the mean error with a few outliers at 4 meters. Frequent maneuvers in the second experiment caused relatively frequent large errors due to the Kalman filter diverging from the ground truth of the tracked node.

6.2 Simulation

The experimental evaluation of our tracking algorithm is somewhat limited due to its complexity and it is a time consuming process. Therefore, we implemented a simulation engine that can evaluate our tracking algorithm using thousands of measurements. The parameters of the simulation engine are:

1. 2D coordinates of infrastructure nodes S_i,
2. the track of the mobile node (a set of timestamped 2D points),
3. the wavelength λ_t of the transmitted signal,
4. σ_m standard deviation of the measurement noise,
5. σ_f standard deviation of the change of the interference frequency ($\widehat{f}$) for consecutive measurements, and
6. the measurement update time t_m.

The engine starts the simulation using the first timestamp recorded in the track and simulates a measurement round every t_m seconds, until the last timestamp. For every measurement round, the location and the velocity of the tracked node is recalculated based on the track data, the relative speeds v_i of the tracked node with respect to S_i nodes are calculated and converted to the frequencies f_i using Eqn. (4). Finally, f_i is corrupted with a normally distributed zero-mean noise with standard deviation σ_m. For the subsequent measurement round, the interference frequency $\widehat{f}$ used in Eqn. (4), is corrupted by a normally distributed zero-mean noise with standard deviation σ_f to simulate its variance over time.

We have simulated both experiments described in the previous section to validate the simulation engine. The parameters of the simulation, $\lambda = 0.67$ cm, $\sigma_m = 0.21$ Hz, $\sigma_f = 1.0$ Hz, and $t_m = 1.7$ sec, were measured experimentally (see Section 5.2). The mean location error of the CNLS-EKF algorithm in simulation was approximately 7% better for the two experiments. A little bit better location accuracy in the simulation is expected because we did not model the routing message-loss in simulation. The speed and the heading accuracy was much better in simulation than in the experiments. We believe that this significant error increase was due to the non-zero time that the maneuver takes in the experiments as compared to the zero time in simulation. We estimate that it takes about a second for a person to stop turn around and accelerate to the desired speed. If we calculate the experimental errors without considering the measurements taken during the maneuvers, the experimental and the simulated errors align within the ground truth measurement error.

6.2.1 Measurements

Simulated data allows us to evaluate our algorithm comprehensively using thousands of measurements under different deployment scenarios and tracks which were generated randomly as follows.

Deployments. We assumed the deployment area to be a 50×30 m rectangle and randomly generated coordinates of the infrastructure nodes from a uniform distribution inside this area. To prevent degenerated cases, we placed four infrastructure nodes in the corners of the area in all deployments discussed in this section.

Tracks. We initialized the tracked node in the center of the deployment area and generated a random 200-point track. The track is defined as a series of timestamped (location,velocity) pairs. The maximum speed of the tracked node v_m and the measurement time t_m are parameters of the simulation engine.

In the following three simulations, we have evaluated the tracking accuracy of the CNLS-EKF algorithm while varying the number of infrastructure nodes, the maximum speed of the tracked node and the tracking update rate.

Simulation 1: The dependance of the tracking accuracy on the number of infrastructure nodes was studied. We generated five different random tracks T_i using $v_m = 2$ m/s and

#receivers	Location (m)	Speed (m/s)	Heading(°)
12	1.4	0.07	5.5
10	1.6	0.08	5.8
8	1.8	0.08	6.2
6	2.3	0.09	8
4	3.3	0.09	8.6

Table 3. Tracking accuracy vs. the number of participating infrastructure nodes. For a given number of receivers, the tracking errors were computed using 5000 randomly generated data points.

$t_m = 1.7$ sec and five different random deployments D_j containing 12 infrastructure nodes. Next, for each deployment D_j, we simulated our tracking algorithm on each of the tracks T_i (collecting 5000 measurements in total). We repeated these simulations by keeping only 4,6,8, or 10 infrastructure nodes out of the 12 original nodes. We show the mean location, speed, and heading accuracy in Table 3.

Simulation 1 shows that the locations of the infrastructure nodes do not have to be specially selected, it is enough if they uniformly cover the tracked area. Note, however, that certain degenerate cases, such as positioning all anchors on a line, can degrade the tracking accuracy of our system.

Speed	Location (m)	Speed (m/s)	Heading(°)
1 m/s	1.6	0.05	5.8
2 m/s	1.8	0.08	6.2
3 m/s	3.6	0.28	7.9
5 m/s	9.0	0.41	13.0

Table 4. Tracking accuracy vs. the maximum speed of the tracked node. For a given maximum speed v_m, the track that the node follows was generated using random speeds from the (0,v_m) interval.

Simulation 2: The CNLS-EKF algorithm was evaluated for different maximum speeds of the tracked node. We used five different deployments which contained eight infrastructure nodes. Also, $t_m = 1.7$ sec stayed constant for the duration of the simulation. We varied the maximum speed of the tracked node v_m, using values $1, 2, 3$ and 5 m/s. For a given maximum speed, we generated five random tracks T_i, and simulated each track with each deployment (again, 5000 measurements were simulated). The performance of the CNLS-EKF algorithm is shown in Table 4. Note that reducing the speed of the tracked node further below 1 m/s would not improve the tracking accuracy by much because the measurement error in the Doppler shift would become relatively large, compared to the measured values.

Simulation 3: Finally, we evaluated the tracking accuracy for different tracking update rates. Similarly to Simulation 2, we kept the deployments constant. Next, we kept the maximum speed constant at $v_m = 5$ m/s and simulated the tracks with update rates of once every $0.4, 0.8, 1.2$, and 1.7 seconds. Results are shown in Table 5.

6.2.2 Discussion

In general, adding more receivers, limiting the maximum speed of the tracked node and increasing the temporal resolution of the collected data help to improve the accuracy of our tracking algorithm.

Update rate	Location (m)	Speed (m/s)	Heading(°)
0.4 sec	1.4	0.15	3.8
0.8 sec	1.8	0.17	7.1
1.2 sec	2.1	0.20	4.8
1.7 sec	9.0	0.41	13.0

Table 5. Tracking accuracy vs. the tracking update rate. For faster update rates, the random tracks were generated with a better temporal resolution which allows for better tracking accuracy.

However, improving the accuracy by increasing the number of infrastructure nodes becomes inefficient relatively soon. In fact, the 8 infrastructure nodes that were deployed in the 1500 m^2 area seems to be a good tradeoff between the tracking accuracy and the required density of the infrastructure nodes. The algorithm quickly becomes inaccurate, if the speed of the tracked node increases, up to a point when the Kalman filter completely diverges from the true location of the tracked node. The tracking accuracy in this case can be improved by increasing the update rate. Improving the update rate to once per 1.2 sec was sufficient to prevent the Kalman filter from diverging when the tracked node moved up to 5 m/s. However, we estimate that on the Mica2 platform, the limitations of the CPU and the low communication speed of the radio chip would prevent us from running the tracking algorithm faster than once every 1 sec. Therefore, our Mica2 implementation will not scale in the case of maneuvering nodes moving at speeds over 5 m/s.

7 Related Work

Accurate location estimation is an essential technology for numerous sensor network applications including tracking people, asset management, and environmental monitoring [13, 11]. Current approaches can be categorized along several dimensions: dedicated network infrastructure vs. existing wireless network infrastructure, self- vs. remote-positioning, anchor-based vs. anchor-free, range-based vs. range-free, centralized vs. localized computation, and signal modality (radio-frequency, infrared, ultrasonic, visual, audio, electromagnetic, laser). Further details can be found in [14] and other surveys that have appeared in the networking, ubiquitous computing, and signal processing literature.

In localization, the position of a node is estimated from static snapshot measurements. Tracking of mobile nodes can be achieved by sequentially estimating the location of a node [1]. Such methods are not accurate because of the unavoidable measurement errors associated with mobility, and further, require real-time performance. Alternatively, mobility allows the use of sampled temporal measurements and motion models that can enhance estimation accuracy and improve sensor localization [11].

The ultrasound-based Cricket location system has been used for tracking mobile nodes in [30]. Each mobile node runs a Kalman filter to estimate its location using distance measurements from the infrastructure nodes. Similarly to our approach, accuracy of the filter is monitored and the

Kalman filter is reset by running least-squares minimization. Since in our case the relative speeds are measured, the least-squares are more sensitive to measurement error, potentially resetting the Kalman filter to a wrong state. Therefore, more careful update of the Kalman filter and the constrained least-squares method were used in our CNLS-EKF algorithm. The accuracy of the Cricket algorithm was evaluated for different speeds and the median error was 15 cm at 1.4 m/s. However, the test area was limited to 3×1.5 meters and the track was limited to an ellipse with no significant changes in the speed of the tracked node.

A distributed localization based on robust quadrilaterals was described in [27]. The method utilizes Cricket's TDOA ultrasound measurements to estimate pairwise distances between the nodes. Weighted least-squares optimization is used to redistribute the measurement errors in the localization process. Since the TDOA methods measure inconsistent distances for mobile nodes, the authors run a simple Kalman filter for each distance measurement before using it in the optimization. In contrast, our CNLS-EKF algorithm uses Kalman filter to process all measurements collected from multiple infrastructure nodes. Similarly to the Cricket approach, the errors reported in this work were in a centimeter range, but the experimental setup was limited to a 2×2 m area.

Radio-frequency (RF) based methods have been previously proposed for tracking mobile users in buildings. RADAR reduces the tracking problem to a sequence of location problems of a nearly stationary user [1]. It combines empirical measurements with signal propagation modeling resulting in 2 to 3 meters accuracy. A few commercial systems such as PinPoint [29] and PalTrack [28], based on TOA and RSS measurements respectively, have been also developed with meter-scale accuracy.

The key idea in tracking mobile nodes using filtering techniques is to include a dynamic model for predicting the position at the next time step. Any model suggested in target tracking using sensor networks is also plausible for this application (see, for instance, [3, 32] and the references therein). Because of the limited computational resources, we use a simple model that assumes constant velocity and direction.

Finally, the Doppler effect has been used extensively to estimate the velocity of tracked objects or to improve the accuracy of tracking systems. However, we are not aware of any sensor network system that uses RF Doppler shifts for tracking mobile objects.

8 Conclusions and Future Work

We have introduced a novel tracking algorithm for wireless sensor networks that utilizes Doppler shifts of the radio signal transmitted by a tracked node. We assumed that a number of stationary infrastructure nodes were deployed around the tracked node and that the tracked node cooperated with the tracking system. We showed that the Doppler shifts can be measured accurately using radio interferometry, allowing the infrastructure nodes to determine the relative speed of the tracked node with 0.14 m/s accuracy using low-cost hardware.

The tracking problem was formulated as an optimization problem with the location and the velocity of the tracked node being the parameters and the measured relative speeds being the constraints of the optimization. We showed that the measurement errors and the non-linearity of our optimization problem can result in poor tracking accuracy in certain cases. The extended Kalman filter (EKF) is a computationally efficient technique that can remove the effects of the measurement errors. We showed that it works well in our case. However, if the node is maneuvering, the accuracy of the EKF becomes poor up to the point of complete divergence of the filter. We suggested to use the constrained non-linear least squares (CNLS) technique to update the state of the EKF if a maneuver was detected. The combined CNLS-EKF algorithm was evaluated both experimentally and in simulation and achieved a location accuracy of $1.3 - 2.2$ m in the best and the worst case, respectively. The accuracy of the speed and bearing estimates were $0.1 - 0.4$ m/s and $7 - 18^{\circ}$, respectively and the location accuracy improvement of our algorithm over the EKF filter was up to 50%.

Currently, only the Doppler shift measurements are running on the sensor nodes. We plan to implement a version of the EKF and the CNLS algorithm on the tracked nodes in the future. The time to route the ranging data to the tracked node as opposed to the central node will improve, thus faster tracking update rates will be possible. In addition, both the EKF and CNLS techniques are computationally efficient, require a small number of steps to converge and a small amount of information to store. Therefore, they are well suited for implementation in sensor networks.

The second area of future work is the scheduling algorithm. In our proof-of-concept implementation, all infrastructure nodes up to two hops away participate in the tracking algorithm. Clearly, this approach wastes resources of the infrastructure nodes and does not allow for the simultaneous tracking of multiple nodes in the same physical area. Also, if the second transmitter is located far from the tracked node, its signal gets attenuated by the time it gets to the tracked node and thus, the interference signal would have a low amplitude. We plan to address these problems by designing a scheduling algorithm which activates and deactivates infrastructure nodes participating in tracking and which selects the second transmitter close to the estimated location of the tracked node.

Finally, our approach is less susceptible to multipath propagation than the radio interferometric technique, since reflections do not change the frequency of the signal. Therefore, additional Doppler shifts can only be introduced by multipath between the mobile node and a single receiver. The phase of the signal, however, can be distorted by multipath propagation between all four transmitter-receiver pairs. Consequently, we plan to evaluate our algorithm in strong multipath environments.

9 References

[1] P. Bahl and V. N. Padmanabhan. Radar: An in-building RF-based user-location and tracking system. *Proc. IEEE INFOCOM*, 2:77584, Mar. 2000.

[2] R. R. Brooks, C. Griffin, and D. S. Friedlander. Self-organized distributed sensor network entity tracking. *The International Journal of High Performance Computing Applications*, 16(3), 2002.

[3] R. R. Brooks, P. Ramanthan, and A. Sayeeed. Distributed target classification and tracking in sensor networks. *In Proc. of the IEEE*, pages 1163–1171, Aug. 2003.

[4] Z. Butler, P. Corke, R. Peterson, and D.Rus. Networked cows: Virtual fences for controlling cows. *In Proc. of WAMES*, 2004.

[5] Chipcon AS, CC1000: Single chip very low power RF transceiver. `http://www.chipcon.com`, 2004.

[6] C.-B. Chang and J. Tabaczynski. Application of state estimation to target tracking. *IEEE Transactions on Automatic Control*, 29–2:98–109, 1984.

[7] P. Dutta, M. Grimmer, A. Arora, S. Bibyk, and D. Culler. Design of a wireless sensor network platform for detecting rare, random, and ephemeral events. *In Proc. of IPSN/SPOTS*, Apr. 2005.

[8] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar. Next century challenges: Scalable coordination in sensor networks. *In Proc. of MobiCom*, 1999.

[9] L. Girod, M. Lukac, V. Trifa, and D. Estrin. The design and implementation of a self-calibrating acoustic sensing platform. *In Proc. of ACM SenSys*, Nov. 2006.

[10] L. Gu, D. Jia, P. Vicaire, T. Yan, L. Luo, A. Tirumala, Q. Cao, J. A. Stankovic, T. Abdelzaher, and B. Krogh. Lightweight detection and classification for wireless sensor networks in realistic environments. *In Proc. of ACM SenSys*, Nov. 2005.

[11] F. Gustafsson and F. Gunnarsson. Mobile positioning using wireless networks. *IEEE Signal Processing Magazine*, 22(4):41–53, July 2005.

[12] G.Wener-Allen, J. Johnson, M. Ruiz, J. Lees, and M. Welsh. Monitoring volcanic eruptions with a wireless sensor networks. *In Proc. of EWSN*, 2005.

[13] M. Hazas, J. Scott, and J. Krumm. Location-aware computing comes of age. *IEEE Computer*, 37(2):95–97, 2004.

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

[15] J. Hill and D. Culler. Mica: a wireless platform for deeply embedded networks. *IEEE Micro*, 22(6):12–24, 2002.

[16] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for networked sensors. *In Proc. of ASPLOS-IX*, 2000.

[17] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, and D. Rubenstein. Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet. *In Proc. of ASPLOS-X*, 2002.

[18] R. E. Kalman. A new approach to linear filtering and prediction problems. *Trans. ASME, Journal of Basic Engineering*, 1960.

[19] B. Kusý, G. Balogh, A. Lédeczi, J. Sallai, and M. Maróti. inTrack: High precision tracking of mobile sensor nodes. *In Proc. of EWSN*, Jan. 2007.

[20] B. Kusý, G. Balogh, P. Völgyesi, J. Sallai, A. Nádas, A. Lédeczi, M. Maróti, and L. Meertens. Node-density independent localization. *In Proc. of IPSN/SPOTS*, Apr. 2006.

[21] B. Kusý, P. Dutta, P. Levis, M. Maróti, A. Lédeczi, and D. Culler. Elapsed time on arrival: a simple and versatile primitive for canonical time synchronization services. *International Journal of Ad Hoc and Ubiquitous Computing*, 2(1), 2006.

[22] B. Kusy, J. Sallai, G. Balogh, A. Ledeczi, V. Protopopescu, J. Tolliver, F. DeNap, and M. Parang. Radio interferometric tracking of mobile wireless nodes. *In Proc. of MobiSys*, 2007.

[23] P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, and D. Culler. Tinyos: An operating system for wireless sensor networks. *Ambient Intelligence, Sprinter-Verlag*, 2005.

[24] X. R. Li and V. P. Jilkov. A survey of maneuvering target tracking: approximation techniques for nonlinear filtering. *Proc. 2004 SPIE Conf. Signal and Data Processing of Small Targets*, 5428-62, 2004.

[25] K. Madsen, H. Nielsen, and O. Tingleff. Methods for nonlinear least squares problems. *Tech. rep.*, 2004.

[26] M. Maróti, B. Kusý, G. Balogh, P. Völgyesi, A. Nádas, K. Molnár, S. Dóra, and A. Lédeczi. Radio interferometric geolocation. *In Proc. of ACM SenSys*, Nov. 2005.

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

[28] PalPtrack, Sovereign Tracking Systems L.L.C. `http://www.sovtechcorp.com/`.

[29] PinPoint, RFTechnologies. `http://www.rft.com/products/pinpoint/`.

[30] A. Smith, H. Balakrishnan, M. Goraczko, and N. Priyantha. Tracking moving devices with the Cricket location system. *In Proc. of MobiSys*, 2004.

[31] N. Xu, S. Rangwala, K. K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin. A wireless sensor network for structural monitoring. *In Proc. of ACM SenSys*, 2004.

[32] F. Zhao, J. Liu, J. Liu, L. Guibas, and J. Reich. Collaborative signal and information processing: An information directed approach. *In Proc. of the IEEE*, 91(8):1199–1209, Aug. 2003.