RF Doppler Shift-Based Mobile Sensor Tracking and Navigation

BRANISLAV KUSÝ
CSIRO ICT Centre
and
ISAAC AMUNDSON, JANOS SALLAI, PETER VÖLGYESI, AKOS LÉDECZI, and XENOFON KOUTSOUKOS
Vanderbilt University

Mobile wireless sensors require position updates for tracking and navigation. We present a localization technique that uses the Doppler shift in radio transmission frequency observed by stationary sensors. We consider two scenarios. In the first, the mobile node is carried by a person. In the second, the mobile node controls a robot. In both approaches the mobile node transmits an RF signal, and infrastructure nodes measure the Doppler-shifted frequency. Such measurements enable us to calculate the position and velocity of the mobile transmitter. Our experimental results demonstrate that this technique is viable and accurate for resource-constrained mobile sensor tracking and navigation.

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

General Terms: Algorithms, Experimentation, Measurement, Theory

Additional Key Words and Phrases: Sensor networks, Doppler effect, tracking, localization, navigation

ACM Reference Format:

Kusý, B., Amundson, I., Sallai, J., Völgyesi, P., Lédeczi, A., and Koutsoukos, X. 2010. RF doppler shift-based mobile sensor tracking and navigation. ACM Trans. Sensor Netw. 7, 1, Article 1 (August 2010), 32 pages. DOI = 10.1145/1806895.1806896 http://doi.acm.org/10.1145/1806895.1806896

This work was supported by ARO MURI grant W911NF-06-1-0076, NSF CAREER award CNS-0347440, NSF grant CNS-0721604, and a Vanderbilt University Discovery grant.

Authors' addresses: B. Kusý, CSIRO ICT Centre, 1 Technology Court, Pullenvale QLD 4069, Australia; email: Branislaw.Kusy@csiro.au; I. Amundson, J. Sallai, P. Völgyesi, A. Lédeczi, and X. Koutsoukos, Institute for Software Integrated Systems, Vanderbilt University, Box 1829, Station B, Nashville, TN 37325; email: {Isaac.Amundson, Akos.Ledeczi, Peter Volgyesi, Xenofon.Koutsoukos}@vanderbilt.edu.

DOI 10.1145/1806895.1806896 http://doi.acm.org/10.1145/1806895.1806896

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 [Estrin et al. 1999; Hill et al. 2000]. WSNs have become powerful tools for the detection, classification, monitoring, and management of objects in the physical world [G.Wener-Allen et al. 2005; Juang et al. 2002; Xu et al. 2004; Butler et al. 2004]. Their demonstrated success stems from their ability to be deployed over wide areas in the environment, close to phenomena of interest. The main strength of WSNs is achieved through distributed collaboration, thus overcoming the limitations of a single sensor. Therefore, the spatiotemporal coordination of sensors becomes important in many WSN applications.

Recently, there has been a concerted effort to integrate mobility into WSNs. Mobile sensor nodes enable dynamic deployment and expand the sensing region [Wang et al. 2005], minimize energy consumption across the network [Gandham et al. 2003], and can maintain connectivity in sparse networks [Shah et al. 2003]. Wireless sensors are now small enough that they can be carried by humans, attached to warehouse packages, and embedded in vehicles. Robotic mote-sized platforms [Bergbreiter and Pister 2003; Dantu et al. 2005; Friedman et al. 2005] have been developed that can sense the environment and react to it by performing tasks throughout the sensing region. Sensor mobility introduces many new challenges. Among these, the need for localization is perhaps the most important. Often, the Global Positioning System (GPS) is cited as the de facto method for sensor node localization. However, GPS has several pitfalls such as cost and power consumption, and cannot be relied on in certain sensor network deployments (i.e., indoors, under dense foliage, in urban environments).

In this article, we consider the problem of accurately keeping track of the location and velocity of mobile sensors continuously over time. Tracking of mobile entities has been an active area of research [Brooks et al. 2002; Li and Jilkov 2004; Chang and Tabaczynski 1984]. Generally, the 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, to be power-efficient, localization should only be performed when a sensor node is in motion. Sensors should be responsible for maintaining their state, including their location, and invoke a localization service only if they detect that their position is changing. In our approach, the initial locations of the sensors are found at deployment time using one of the computationally more expensive methods, such those described as in Girod et al. [2006], Maróti et al. [2005], and Moore et al. [2004]. When a node detects that its location has changed, it notifies stationary sensor nodes deployed in the vicinity, which then participate in the localization process. The location and velocity information is sent back to the mobile node, in order to keep its state estimate accurate at all times.

In Maróti et al. [2005] and Kusý et al. [2006a], a radio interferometric ranging technique was proposed for the precise localization of stationary nodes. This technique was extended to track mobile nodes in Kusý et al. [2007a]. The

approach utilizes interfering radio signals transmitted by two nodes simultaneously. The relative phase offset of the interference signal at two different receivers is a linear combination of the distances between the four nodes involved. We use a similarly constructed interference signal, but measure its frequency rather than its phase. Localization is accomplished by observing the Doppler shift in frequency that occurs when the source of a transmitted signal is moving relative to an observer. Because Doppler shift is determined by the relative velocity of the transmitter and receiver, position *and* velocity of a mobile entity can be simultaneously determined using a priori knowledge of the transmitted frequency and the frequencies observed by stationary sensor nodes at known positions.

Our approach allows for simpler and faster localization because each of the receivers measure the Doppler-shifted frequencies only, whereas relative phase measurements need to be carried out by pairs of receivers in Kusý et al. [2007c]. In addition, Doppler shifts are measured at a single carrier frequency, whereas many different frequencies are required to obtain accurate ranging data from phase measurements in Kusý et al. [2007b].

This article presents several new contributions to the field of mobile WSN tracking and navigation.

(1) We develop an algorithm that estimates the position and velocity of a mobile node from RF Doppler-shifted frequency measurements. The algorithm can be used differently depending on whether the node can control its own movement. For the case in which the sensor moves under the influence of an external force, such as when carried by a person, the primary concern is tracking the movement of the node and keeping its location known at all times. We refer to this scenario as *dTrack*. For the case in which the node controls its movement, such as a robot, the position and velocity obtained from the algorithm are used as control feedback, which keeps the mobile node from deviating from its target trajectory. We refer to this scenario as *dNav*.

(2) For dTrack, we develop a constrained nonlinear least-squares/extended Kalman filter optimization algorithm (CNLS-EKF), which detects and compensates for maneuvers. Generally, an EKF can be used alone for tracking mobile sensors under a constant-velocity assumption. However, it is a well-known problem that the EKF fails to accurately track sudden maneuvers of mobile objects [Chang and Tabaczynski 1984]. To improve the tracking accuracy in this case, we develop the CNLS-EKF algorithm, and show that this improves the EKF accuracy by greater than 50%.

(3) For dNav, using our localization algorithm, we develop a navigation technique for mobile robots in which the observed Doppler-shifted frequency is used as feedback to control the movement of the mobile node. The CNLS algorithm is unnecessary because the robot is aware of any maneuvers it makes. We show that, by feeding this process input to the EKF, the mobile node is able to accurately navigate a sensing region without experiencing localization error due to such maneuvers.

(4) We implement our dTrack and dNav algorithms on a real-world resource-constrained WSN platform. For dTrack, radio interferometry and routing is implemented on XSM motes; however, the localization algorithm runs on a PC base station. For dNav, all processing runs on two robot-mounted motes within the control-loop, and requires no additional PC processing.
(5) We performed several experiments using the dTrack and dNav implementations in order to demonstrate the viability and accuracy of the tracking and navigation algorithms. We present our results, which are highly accurate, and have superior runtime and memory complexity compared with previous methods [Kusý et al. 2007b; Amundson et al. 2008].

The remainder of this article is organized as follows. Section 2 describes existing methodologies for localization and navigation. We state the problem and discuss the proposed technique used to measure Doppler shifts in Section 3. In Section 4, we present the description of the CNLS and EKF techniques and apply them to our tracking problem (dTrack). Section 5 applies our algorithm to mobile sensor navigation (dNav) and discusses the differences from the tracking case. Section 6 concludes the article.

2. RELATED WORK

Accurate location estimation is an essential technology for numerous sensor network applications including tracking people, asset management, and environmental monitoring [Hazas et al. 2004; Gustafsson and Gunnarsson 2005]. Current approaches can be categorized along several dimensions: dedicated network infrastructure versus existing wireless network infrastructure, self- versus remote-positioning, anchor-based versus anchor-free, range-based versus. range-free, centralized versus localized computation, and signal modality (radio-frequency, infrared, ultrasonic, visual, audio, electromagnetic, laser). Further details can be found in Hightower and Borriello [2001] 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 [Bahl and Padmanabhan 2000]. Such methods are not accurate because of the unavoidable measurement errors associated with mobility, and, furthermore, 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 [Gustafsson and Gunnarsson 2005].

The ultrasound-based Cricket location system was used for tracking mobile nodes in Smith et al. [2004]. Each mobile node runs a Kalman filter to estimate its location using distance measurements from the infrastructure nodes. 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 m and the track was limited to an ellipse with no significant changes in the speed of the tracked node.

A distributed localization technique based on robust quadrilaterals was described in Moore et al. [2004]. The method utilizes Cricket's time difference of arrival (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 ran a simple Kalman filter for each distance measurement before using it in the optimization. In contrast, our CNLS-EKF algorithm uses the Kalman filter to process all measurements collected from multiple infrastructure nodes. Similarly to the Cricket approach, the errors reported in this work were in the centimeter range, but the experimental setup was limited to a 2×2-m area.

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 [Bahl and Padmanabhan 2000]. It combines empirical measurements with signal propagation modeling, resulting in an accuracy of approximately 3 m. A few commercial systems such as PinPoint[1] by RFTechnologies and PalTrack[2] by Sovereign Tracking Systems L.L.C. based respectively on time of arrival (TOA) and received signal strength (RSS) measurements, have also been developed with meter-scale accuracy.

NavMote [Fang et al. 2005] is a pedestrian dead reckoning system that uses accelerometer and compass data to derive the location and heading of the mobile node. The data is stored locally until the mobile node comes in range of the network, at which time it is transferred wirelessly to a resource-heavy information server for processing.

The mobile localization problem has also been extensively studied in robotics, and some methods have been adapted for sensor networks. A sequential Monte Carlo localization method which exploits node mobility was presented in Hu and Evans [2004]. The method is based on whether a node is within radio range of other nodes, and performs well for large and dense sensor networks. LaSLAT is a Bayesian framework for simultaneous localization and tracking [Taylor et al. 2006] which employs a mobile sensor for localizing the other nodes in the network. LaSLAT uses ultrasound and acoustic ranging with centimeter 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, e.g., Brooks et al. [2003], and Zhao et al. [2003], 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 to estimate the velocity of tracked objects or to improve the accuracy of tracking systems [Chan and Jardine 1990; Chan and Towers 1992]. In recent WSN research, Doppler shifts generated by rotating beacons [Ledeczi et al. 2008; Chang et al. 2008] were used for target localization, and, in Amundson et al. [2009], Doppler shifts were used in

[1]http://www.rft.com/products/pinpoint/.
[2]http://www.sovtechcorp.com.

conjunction with wheel encoder data to determine pairwise angular separation between landmarks for mobile sensor navigation. We are not aware, however, of any sensor network system that only uses RF Doppler shifts for tracking mobile objects.

3. LOCALIZATION USING RF DOPPLER SHIFTS

In this section, we present our technique for localization using RF Doppler shifts. We make the following assumptions regarding the design of our localization service: (a) a number of infrastructure nodes are deployed at known locations in the area of interest, (b) mobile nodes cooperate with the infrastructure nodes to find their location and velocity vectors, and (c) the hardware capabilities of both the infrastructure and tracked nodes are limited. Consequently, the design of our localization service mandates relatively low sampling rates and algorithms with limited memory, computation, and communication requirements.

3.1 Localization Approach

A well-known phenomenon that is observed when objects move relative to each other is the Doppler effect. The Doppler effect law states that, when an object transmits a signal while moving 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 relative velocity of the transmitter and receiver. We take 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 mobile 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 a mobile node T, which is moving through a region where six infrastructure nodes are deployed ($S_i, i = 1 \cdots 6$). The velocity vector $\overrightarrow{v}$ of T is shown at two different locations in the two figures. The magnitude of the Doppler shift observed at node S_i depends on the relative speed of T and S_i, which can be found by projecting the velocity $\overrightarrow{v}$ onto the unit position vector $\overrightarrow{S_iT}$. 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 projection 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, by measuring a sufficient number of relative speeds, both the location and the velocity vector of the mobile node can be found accurately.

Our approach follows the general three-phase structure of many existing localization and tracking algorithms [Brooks et al. 2002; Moore et al. 2004; Girod et al. 2006]:

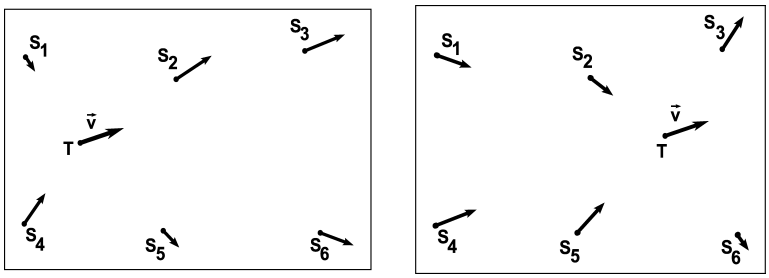

Fig. 1. The Doppler effect gives us information on both the velocity and the location of a moving object.

—*Coordination phase* Infrastructure nodes are notified to participate in the localization of a node in a certain region. Both the timeframes and the local coordinate systems of the participating nodes are synchronized to enable the data fusion of spatially and temporally distributed measurements. In addition, other initialization tasks are performed at this time.

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

—*Localization phase* Nonlinear optimization and filtering techniques are used to smooth the measurement noise by combining the ranging data measured at multiple infrastructure nodes at subsequent time steps. The movement of the mobile node can be predicted and the infrastructure nodes participating in the localization can be activated or deactivated accordingly.

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

3.1.1 *Coordination Phase.* We assume that the mobile node cooperates with the localization system. However, we still need to identify the infrastructure nodes that will participate in the localization. This is achieved by having the mobile node broadcast a localization request. All infrastructure nodes that receive the request will participate in the localization. We assume that the positions of the infrastructure nodes are presurveyed 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 localization system. However, the mobile node and the participating infrastructure nodes need to be time-synchronized with relatively high accuracy. This is required by the ranging method that we use, as well as to allow for the correct fusion of the ranging data. It was shown in Kusý et al. [2006b] 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 localization-request message broadcast by the mobile node is used as the

synchronization point, allowing us to achieve time synchronization accuracy on the order of microseconds.

3.1.2 *Measurement Phase.* The mobile node transmits a signal and multiple infrastructure nodes measure the Doppler-shifted frequency. 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 greater detail in Section 3.2.

3.1.3 *Localization Phase.* An EKF is used to estimate the location and velocity of the mobile node from the Doppler-shifted frequency measurements obtained in the measurement phase. With dTrack, when a maneuver is detected we update the Kalman filter state by running CNLS optimization on the last set of collected measurements. With dNav, we know the maneuver being made, and provide this information to the Kalman filter directly. More details on how both the EKF and CNLS techniques are applied to our localization problem can be found in Sections 4 and 5.

3.2 Measuring Doppler Shifts

Measuring the frequency of a given signal with sufficient accuracy becomes a challenge when using resource-constrained hardware with a limited sampling rate. A popular and efficient way to determine the frequency of a signal is frequency domain analysis. However, it was shown in Gu et al. [2005] that computing the frequency spectrum by FFT is prohibitively expensive given our typical platforms. In particular, it would take approximately 15 s to calculate a 512-point FFT using an 8-MHz processor typically available in many 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 a smaller Doppler shift, which in turn requires greater measurement accuracy.

Sensor networks are well suited for two types of 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 resource-constrained WSN hardware. Radio signals are favored over acoustic signals because sensor nodes are equipped with the radio transceivers for wireless communication, and therefore no additional hardware is required. Moreover, radio transmission is unobtrusive and less prone to interference from the environment. The Doppler shift observed at the typical carrier radio frequencies (400 MHz–2.4 GHz) is 1.3–8 Hz per 1 m/s of velocity. However, these frequencies are too high to be analyzed directly using inexpensive mote radio hardware.

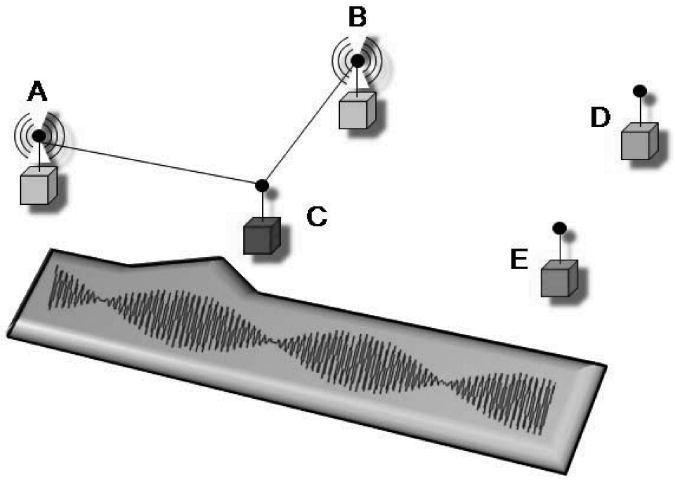

Fig. 2. Two transmitters A and B transmit at the same time at two close frequencies. The interference signal is observed by receivers C, D, and E.

The Radio Interferometric Positioning System (RIPS) was proposed in Maróti et al. [2005] to derive location information by analyzing the phase of radio signals with low cost hardware. In this approach, two nodes simultaneously transmit pure sine waves at slightly different frequencies, 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 signals, then the resulting interference signal has a frequency of $(f_a + f_b)/2$ and a low-frequency envelope of $|f_a - f_b|$. Figure 2 shows an example of the interference signal and its low-frequency beats at node C. The theoretical radio interference model was developed to link the phase difference of the interference signal at two receivers to the so called *q-range* quantity—a linear combination of distances between the two transmitters and two receivers. By measuring multiple 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 low enough; and (c) it allows for a large range (more than twice that of the communication range) because it is the phase of the radio signals that is analyzed, rather than the amplitude. Furthermore, because the measurement range is significantly larger than the data communication range, all nodes that receive the coordination message to perform the localization will be able to make accurate measurements, and will not experience degradation of the RSSI signal. Other RSS-based localization techniques suffer degradation when using signals originating near the boundary of the communication radius, impacting the scalability of those techniques.

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 as well Kusý et al. [2007c]. However, the Doppler shifts alone could not be used for tracking in this approach as they did not carry enough information to calculate the positions of the tracked nodes. This was because the tracked nodes were assigned to participate in the radio interferometric measurements 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 not enough to find its position.

In Kusý et al. [2007b], we assigned the mobile node to be a transmitter. Thus, a number of infrastructure nodes were able collect sufficiently many measurements to determine the location of the mobile node from the Doppler shifts alone. In this scenario (Figure 3), the mobile node T transmits an unmodulated sine wave at frequency f_t, and an infrastructure node A transmits 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. Because infrastructure nodes do not move, and the signal transmitted by A is not Doppler-shifted, the measured envelope frequency f_i of the interference signal at node S_i is given by

$$f_i = f_t - f_a + \Delta f_t^i. \tag{1}$$

Equation (1) allows us to calculate the Doppler shift measured at node S_i, and, consequently, the relative speed of the mobile node. This approach differs from the RIPS algorithm in that we do not measure the phase of the interference signal. A disadvantage is that we lose the range information which can be deduced from the relative phase of two receivers. However, a simpler localization algorithm can be implemented: (a) measuring the frequency of the interference signal allows for faster time-domain analysis of the signal, (b) frequency-based localization is not affected by the modulo 2π ambiguity that arises in RIPS phase measurements, 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 [Kusý et al. 2006a], which significantly increased the measurement time.

We express the Doppler-shifted frequencies measured by the infrastructure nodes as a function of the position and velocity of the mobile node. Suppose that $\overrightarrow{v}$ is the velocity of the mobile node T and $\overrightarrow{u_i} = \overrightarrow{S_iT}/\|S_iT\|$ is the unit length vector pointing from sensor S_i to T (See Figure 4). The relative speed of S_i and T can be defined as the following dot product:

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

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

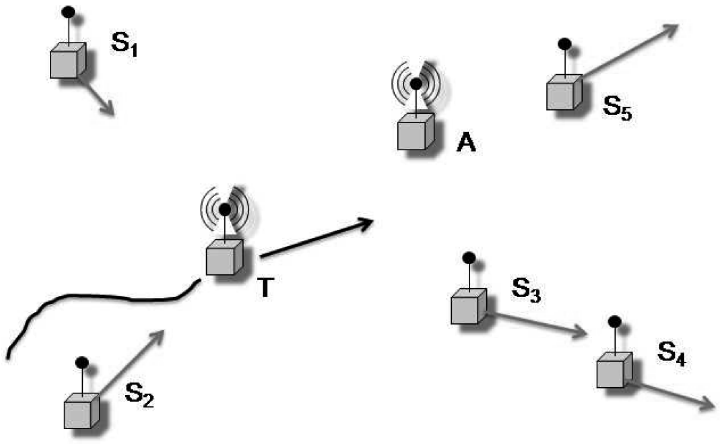

Fig. 3. The mobile node T is being localized by five infrastructure nodes S_i. One other node A cotransmits with T. The receivers S_i calculate the relative speed of T from the measured the Doppler shifts.

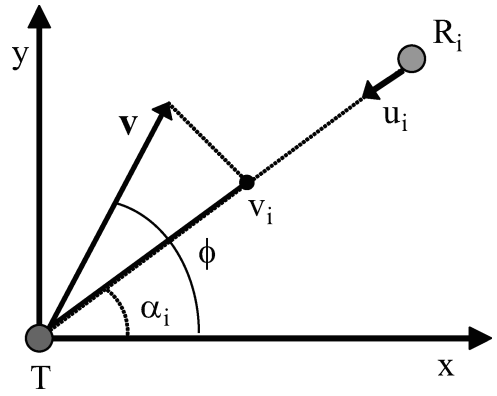

Fig. 4. Mobile node T having velocity v transmits a signal. Sensor S_i measures the Doppler shift of the signal, which depends on v_i, the relative speed of T and S_i.

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, then the observed frequency is $f' = (1 - v/c)f$. Therefore, the Doppler shift is expressed as

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

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

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

Equation (4) allows us to compute the relative speed of the mobile node T and the node S_i if the difference of the two transmitted frequencies $\widehat{f}$ is known.

Unfortunately, estimating $\widehat{f}$ with sufficient accuracy becomes a problem when using low-cost radio transceivers. In previous work [Kusý et al. 2007b; Amundson et al. 2008], we treated $\widehat{f}$ as an unknown parameter in our localization algorithm. However, if we take the pairwise difference of frequency measurements, $\widehat{f}$ is subtracted out. This provides an additional benefit of reducing the run-time complexity of our localization algorithm. Hence, we use the pairwise difference of frequency measurements as our observation, as described in the next section.

3.3 Localization as an 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 mobile node. Therefore, we define our parameter vector X as

$$X = \begin{bmatrix} x & y & v_x & v_y \end{bmatrix}^{\mathsf{T}}.$$

The parameter vector X is related to an observation vector z. We formalize this relation through a function H such that

$$z = H(X).$$

From Equation (4), we see that f_i is a function of $\widehat{f}$, which is an unknown value and highly variable for low-cost radios due to the short-term stability of the quartz crystal. We would therefore like to solve X without having to deal with $\widehat{f}$. We can do this by taking the pairwise difference of the frequency measurements. Assuming n infrastructure nodes measure the Doppler-shifted radio signal, this yields $\binom{n}{2}$ pairwise differences. However, we only require $n-1$ pairwise differences since no additional information can be acquired by using more. Therefore, we use $n-1$ observations z_i. We formally define an observation as

$$z_i = f_{i+1} - f_i = \left(\widehat{f} - \frac{v_{i+1}}{\lambda_t}\right) - \left(\widehat{f} - \frac{v_i}{\lambda_t}\right) = \frac{v_i - v_{i+1}}{\lambda_t}$$

and the observation vector z as

$$z = \begin{bmatrix} z_1 & z_2 & \cdots & z_{n-1} \end{bmatrix}^{\mathsf{T}}.$$

The relative speed v_i of the nodes T and S_i is simply the projection of $\overrightarrow{v}$ onto $\overrightarrow{u_i}$ (Equation (2)). Taking the difference of two such relative speeds, we get

$$v_i - v_{i+1} = \overrightarrow{v} \cdot (\overrightarrow{u_i} - \overrightarrow{u_{i+1}}).$$

Finally, $\overrightarrow{v} \cdot (\overrightarrow{u_i} - \overrightarrow{u_{i+1}})$ can be calculated using the velocity components (v_x, v_y) and coordinates (x, y) of the mobile node, and known quantities λ and the (x_i, y_i) and (x_{i+1}, y_{i+1}) coordinates of nodes S_i and S_{i+1}, respectively. We now define our measurement function (for $i = 1, \ldots, n-1$) as

$$\begin{aligned} H_i(X) \;=\; & v_x \left(cos\left(tan^{-1}\left(\frac{y_i - y}{x_i - x}\right)\right) - cos\left(tan^{-1}\left(\frac{y_{i+1} - y}{x_{i+1} - x}\right)\right)\right) \\ & - v_y \left(sin\left(tan^{-1}\left(\frac{y_i - y}{x_i - x}\right)\right) - sin\left(tan^{-1}\left(\frac{y_{i+1} - y}{x_{i+1} - x}\right)\right)\right). \end{aligned} \quad (5)$$

We estimate the parameters by finding $X \in \mathcal{R}^4$ such that $\|H(X) - z\|$ is minimized. Note that components of our objective function H are nonlinear functions, requiring the use of nonlinear optimization methods.

3.4 Nonlinear Least Squares

Nonlinear optimization techniques typically start with an initial approximation of the parameter vector 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 ([Madsen et al. 2004]) which is based on a linear approximation of the objective function in the neighborhood of X. For example, our objective function $H(X) - z$ is linearized by Taylor expansion as

$$H(X + \Delta) - z \simeq l(\Delta) = H(X) - z + J(X)\Delta,$$

where $J \in \mathcal{R}^{n\times 4}$ is the Jacobian of $H(X) - z$.

The detailed description of the Gauss-Newton algorithm follows. Assuming the ith parameter vector X_i is given,

(1) calculate Jacobian $J_i = J(X_i)$ and linearize the objective function around X_i (denoted by $l_i(\Delta)$),
(2) calculate a local minimizer Δ_i of the function $l_i(\Delta)$, and
(3) set $X_{i+1} = X_i + \alpha\Delta_i$, where α is the step length influencing the convergence of the method. Stop if Δ_i is small.

An additional problem is that $H : \mathcal{R}^4 \to \mathcal{R}^n$ is a vector function. Nonlinear least-squares (NLS) techniques define a new objective function $\mathcal{H} : \mathcal{R}^4 \to \mathcal{R}$ as

$$\begin{aligned}\mathcal{H}(X) &= \frac{1}{2}\sum_{i=1}^{n}(H_i(X) - z_i)^2 \\ &= \frac{1}{2}(H(X) - z)^{\mathsf{T}}(H(X) - z). \end{aligned} \tag{6}$$

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

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

Consequently, the formula for the gradient of L is

$$L'(\Delta) = J(X)^{\mathsf{T}}(H(X) - z) + J(X)^{\mathsf{T}}J(X)\Delta,$$

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

$$\Delta_{\min} = (J(X)^{\mathsf{T}}J(X))^{-1}J(X)^{\mathsf{T}}(z - H(X)). \tag{7}$$

A more detailed derivation of this equation can be found in Madsen et al. [2004].

3.4.1 *Constrained Optimization.* In tracking, we are often able to constrain the area where the tracked node is located. Therefore, applying *constrained* nonlinear least squares may yield more accurate results. One way to solve the constrained optimization is to modify the objective function by adding

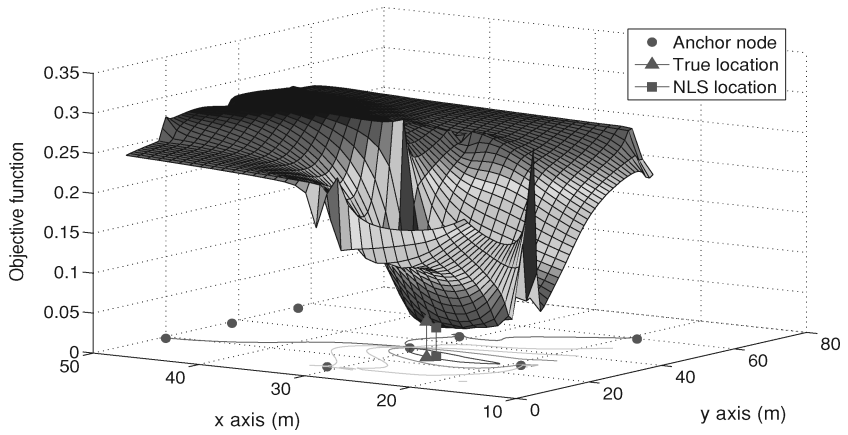

Fig. 5. The objective function $\mathcal{H}$ minimizes the least-squares error in the observation vector. Due to measurement errors, the global minimum of $\mathcal{H}$ is not at the true location of the tracked node.

a barrier function to it, introducing zero penalty inside the region of interest and positive penalty outside of it.

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 $b(X)$ can be defined as

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

where X 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 is similar to the NLS algorithm, except it optimizes the function $\mathcal{H}(X) + b(X)$. The derivatives used in the NLS algorithm need to be adjusted by adding the term $B(X) = \dfrac{\partial b(X)}{\partial X}$ to $L'(\Delta)$. Consequently, Equation (7) becomes

$$\Delta_{\min} = (J(X)^{\mathsf{T}} J(X))^{-1}[J(X)^{\mathsf{T}}(z - H(X)) + B(X)]. \tag{8}$$

3.4.2 *Problems with NLS optimization.* Nonlinear least-squares optimization may fail depending on the starting point X_0 and the measurement errors that corrupt the observation vector z. 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^2 area and measured the Doppler-shifted frequency of the transmitted signal. In Figure 5, we show the positions of the infrastructure nodes as black dots and the position of the transmitter as a triangle. The function plotted in the figure is obtained by finding the minimum value of $\mathcal{H}(X)$ for the fixed coordinates (x, y) (i.e., finding the best fit for the two remaining parameters). Figure 6 shows the best-fit velocities found at a given position.

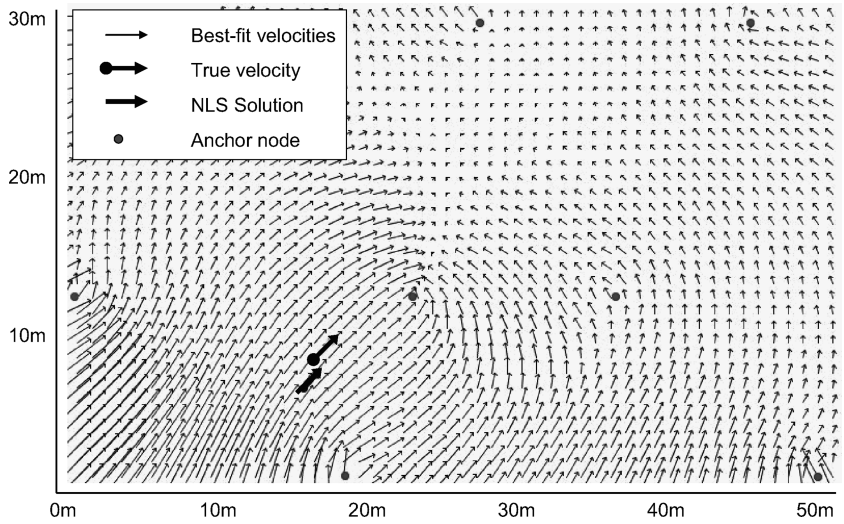

Fig. 6. The best-fit velocities of the tracked node that minimize the function $\mathcal{H}$ in Figure 5 are shown.

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 position of the tracked node in certain cases, we need to find alternative 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 Figure 6, is accurate in a relatively large area around the true location of the tracked node. This allows us to use the velocities calculated by the CNLS algorithm with higher confidence than the position estimates.

3.5 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 the motion model, and update the state based on the new observations. The Kalman filter is a widely used technique for estimating the state of a dynamic system based on noisy measurement data. We use the *extended* Kalman filter [Kalman 1960; Welch and Bishop 2004] because we are dealing with the nonlinear relationship between observed frequencies and relative velocity (Equation (5)). The EKF linearizes the estimation about the current state by applying the partial derivatives of

the process and measurement functions. These functions take the form

$$X_k = F(X_{k-1}, u_{k-1}) + w_{k-1}, \tag{9}$$
$$z_k = H(X_k) + \zeta_k, \tag{10}$$

where $X = \begin{bmatrix} x & y & v_x & v_y \end{bmatrix}^{\mathsf{T}}$ is the system state, z is the measurement vector, u is the process input (0 for the dTrack scenario), w is the process noise with covariance Q, ζ is the measurement noise with covariance R, and k is the current timestep. Both the process noise w and the measurement noise ζ are assumed to have "white noise" properties with zero mean and their covariance matrices are determined experimentally. The state transition vector function F is

$$F = \begin{bmatrix} x_{k-1} + \Delta t \cdot v_{x,k-1} \\ y_{k-1} + \Delta t \cdot v_{y,k-1} \\ v_{x,k-1} \\ v_{y,k-1} \end{bmatrix},$$

where Δt is the time elapsed since the calculation of the previous state. The nonlinear observation function H is defined in Equation (5).

The EKF recursively estimates the system state in two phases. The first phase predicts the state at the current time step based on the state at the previous time step and the current process input. The second phase adjusts the prediction with actual measurement data obtained during the current time step. In addition, an error covariance matrix, P, is maintained, which is a measure of the accuracy of the estimated state, and is used to update the Kalman gain. Formally, these two phases are represented as the following:

(1) *Prediction phase*:

$$\begin{aligned} \hat{X}_k^- &= F(\hat{X}_{k-1}, u_k), \\ P_k^- &= A_{k-1} P_{k-1} A_{k-1}^T + Q, \end{aligned} \tag{11}$$

where $\hat{X}_k^-$ and P_k^- are the a priori state and covariance estimates for the current time step k, and A_{k-1} is the Jacobian of F with respect to X_{k-1}.

(2) *Update phase*:

$$\begin{aligned} K_k &= P_k^- J_k^T (J_k P_k^- J_k^T + R)^{-1}, \\ \hat{X}_k &= \hat{X}_k^- + K_k (z_k - H(\hat{X}_k^-)), \\ P_k &= (I - K_k J_k) P_k^-, \end{aligned} \tag{12}$$

where K is the Kalman gain, J is the Jacobian of H with respect to X, R is the covariance of the measurement noise, and I is the identity matrix.

4. TRACKING

4.1 Problems with the Kalman filter

We applied the EKF to dTrack 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 in which the tracked node changes its direction

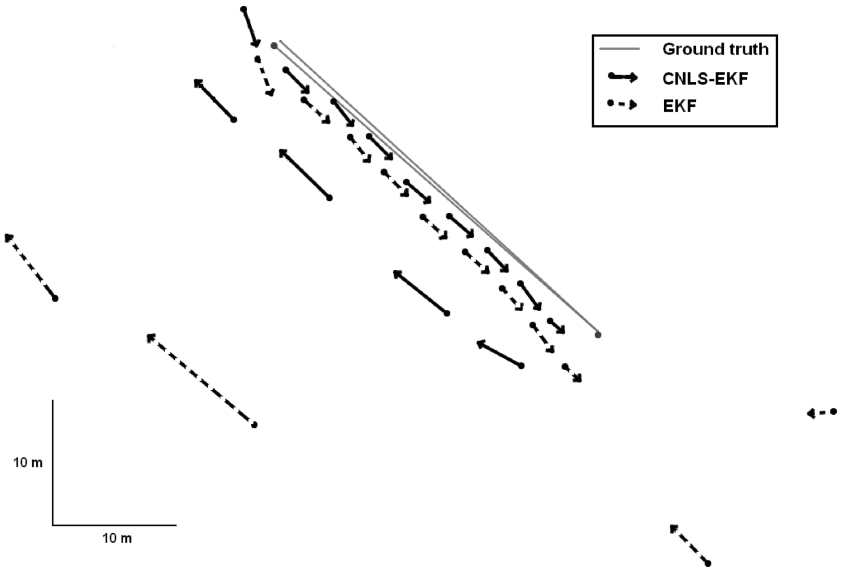

Fig. 7. The EKF fails if the tracked node makes sudden maneuvers. The CNLS-EKF outperforms the EKF in the maneuvering case, while keeping good performance in the nonmaneuvering case.

significantly, for example by 180°, are potentially the most severe. We illustrate this problem in Figure 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 illustrated in the figure, the EKF location error is quite large.

The divergence of the EKF in tracking maneuvering nodes is a well-known problem. A simple solution is to increase the process noise covariance 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. The EKF update algorithm can also be executed iteratively, to better approximate the error function locally. The disadvantage of these techniques is their higher computational complexity, as the EKF prediction and update phases need to be executed multiple times per observation.

4.2 Solving the Maneuvering Case

By combining the EKF and the CNLS techniques, we can significantly improve the tracking accuracy in the maneuvering case, while maintaining the good performance of the EKF in the nonmaneuvering case. We refer to the combined algorithm as *CNLS-EKF*. 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 z, calculate the new EKF state $\mathcal{S} = (X, P)$ using Equations (12) and (13).
(2) If a maneuver is detected, find a new system state X^* by running the CNLS optimization initiated at X.
(3) Use the new system state X^* to update the EKF state $\mathcal{S}$.

4.2.1 *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 error in predicted location (see Figure 6).

4.2.2 *CNLS-Based EKF Update.* Let X^* be the CNLS solution. The EKF state $\mathcal{S} = (X, P)$ is updated by running the linear Kalman filter update algorithm using X^* as the observation vector and the identity matrix I_4 as the measurement matrix. In particular, the Kalman gain K'_k and the updated state $\mathcal{S}' = (X', P')$ are

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

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

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

where ρ is a scaling factor. The base of the exponential function is higher for the position coordinates than the velocity coordinates because the velocity estimates are less prone to measurement errors (see Figure 6). The scaling factor ρ allows us to fine-tune the weight of the CNLS solution.

The improvement of the CNLS-EKF algorithm over using only the EKF can be seen in Figure 7. The tracking accuracy improves by as much as 50% (see Section 4.4).

4.3 Implementation

We have implemented the dTrack system using the TinyOS operating system [Levis et al. 2005]. Our hardware platform is the low-power, battery-operated wireless ExScal mote (XSM) [Dutta et al. 2005], which is a Mica2 [Hill and Culler 2002] compatible mote enclosed in a weather-proof packaging. The most important criterium for our platform choice is the functionality provided by the onboard CC1000 radio chip, which allows for the implementation of the

radio interferometric technique. For our tracking implementation, the CNLS-EKF algorithm runs on a PC-class base station.

4.3.1 *Creating the Interference Signal.* The interference signal is created when two nodes transmit simultaneously at different frequencies. If the difference between the two frequencies is small, the resulting signal has a low-frequency envelope, which can be observed at the Received Signal Strength Indicator (RSSI) pin of the radio chip. Since the frequency of the envelope is equal to 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 beat 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 us to accurately set the transmission frequency. We have observed a 10 parts-per-million (ppm) clock oscillator error, corresponding to a ±4-kHz error, 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 beat frequency f_i, according to Equation (1). For this reason, we were not able to treat $\widehat{f} = f_t - f_a$ as a known quantity 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 (Equation (4)). However, a 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 uncalibrated (thus arbitrarily shifted) radio frequency, while the second transmitter sweeps a relatively large frequency band around this frequency. A nearby receiver observes the interference of the two waves and can determine when the two nodes transmit at the same frequency, allowing 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 Kusý et al. [2007a] for more details).

4.3.2 *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 predefined 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- to 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. This algorithm runs online and does not require any postprocessing.

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 six times. For these parameters, the overall measurement time is 0.4 s and the standard deviation of the observed measurements is 0.21 Hz.

4.4 Evaluation

To evaluate our tracking algorithm, we ran outdoor experiments and calculated the accuracy of the position and velocity estimates of the mobile node.

4.4.1 *Experimental Setup.* We utilized eight infrastructure nodes that measured the Doppler effect and deployed them in a 50×30-m^2 area (see Figure 8). Since two nodes are required to transmit in our approach, an additional infrastructure node was used to cotransmit 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 a two-hop diameter for the duration of the experiment.

It is difficult to estimate the ground truth for a mobile 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. 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 line, using the measurement time as the interpolation coefficient. This process resulted in errors less than 1 m, 0.2 m/s, and 5° in the location, speed, and heading estimates, respectively.

4.4.2 *Experimental Results.* We ran experiments that evaluate our algorithm in two cases: (a) the tracked object moves along straight lines for

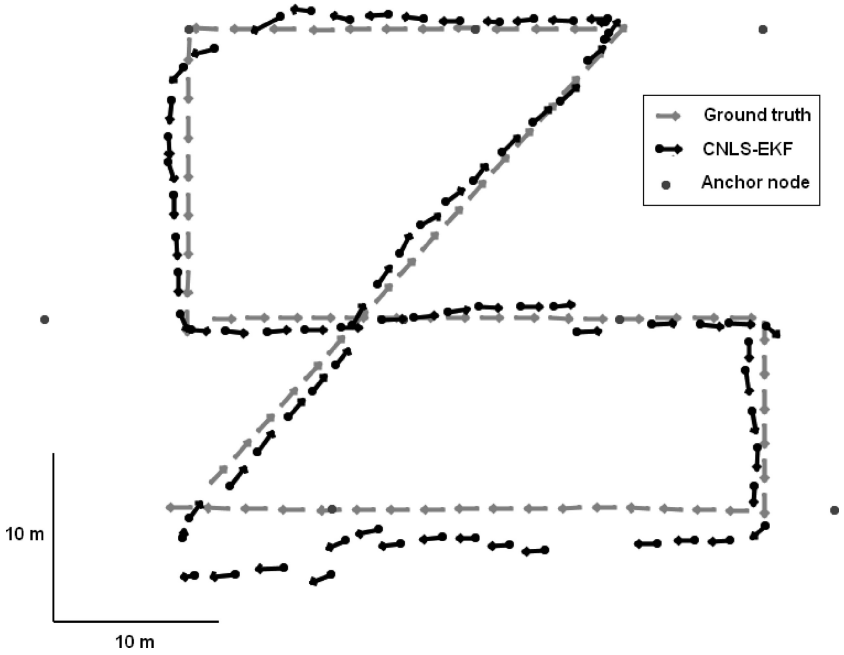

Fig. 8. Experimental evaluation of the CNLS-EKF algorithm. Eight anchor nodes are shown as black dots. The position and velocity of the tracked node are shown as a black dot and an arrow, respectively. The ground truth is shown in gray.

Table I. The Mean Errors were Calculated Based on 73 Measurements Collected During Experiment 1. The Improvement Over the EKF was Modest

Mean error	Position	Speed	Heading
EKF algorithm	1.56 m	0.14 m/s	9.02°
CNLS-EKF algorithm	1.33 m	0.13 m/s	7.89°
Improvement over EKF	14%	7%	12%

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 Figure 8. The tracked node was moving at a mean speed of 1.3 m/s, varying at most by 0.2 m/s. We ran the CNLS-EKF algorithm on this data, setting the process noise covariance matrix Q to 0.5, the measurement noise covariance matrix R to 0.2, and the radius r of the barrier function $\mathbf{b}(\mathbf{x})$ to 3 m. The mean location, speed, and heading errors can be found in Table I. We also ran the EKF alone on the same data set and found that the CNLS-EKF achieves only a modest accuracy improvement in this case.

—*Experiment 2.* We evaluated the scenario that violated the assumption of the linear dynamics of the tracked node, on which the Kalman filter was

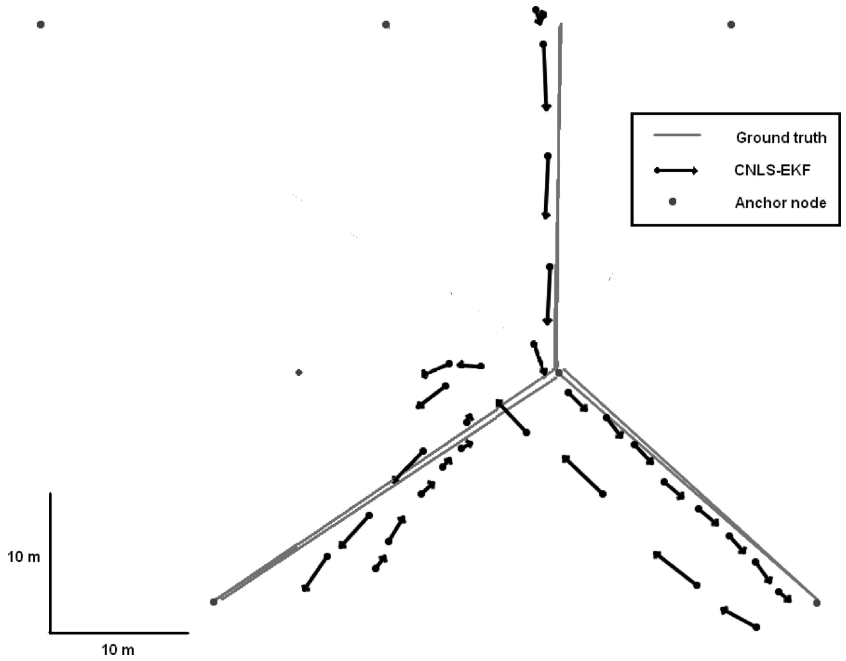

Fig. 9. We show the worst-case situation for the Kalman filter: in a star-like topology, the tracked node moved slowly away from the center and then fast toward the center of the star. Results for only three out of seven tracks are shown for clarity.

Table II. We Collected 92 Measurements in Experiment 2. The Improvement Over the EKF was More Significant, Especially in Location Estimation

Mean error	Position	Speed	Heading
EKF algorithm	10.43 m	0.56 m/s	23.53°
CNLS-EKF algorithm	3.90 m	0.42 m/s	16.83°
Improvement over EKF	62%	25%	28%

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 Figure 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, toward the center. We measured 92 data points in this case and show the results in Table II. The improvement over the EKF with no maneuver correction was more significant this time.

4.5 Analysis of Experimental Results

Both the EKF and CNLS-EKF algorithms perform well for dTrack, if the dynamics of the tracked node is consistent with the EKF model. If the tracked node maneuvers infrequently, the EKF is able to converge to the true location

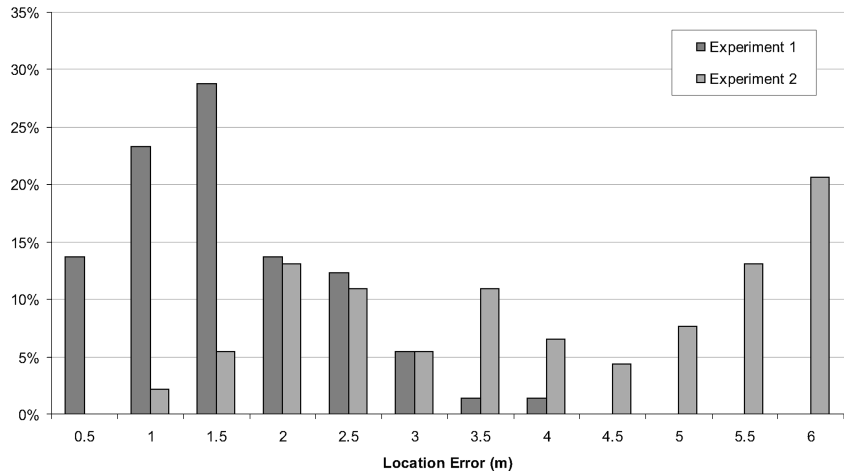

Fig. 10. Distribution of the location estimation errors of the CNLS-EKF algorithm for both experiments.

of the tracked node relatively fast after the maneuver. Consequently, the small additional error of the filter is averaged out for the whole track.

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 and a better overall localization accuracy. Notice that the velocity estimates have approximately the same errors for both the EKF and the 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 about 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 Figure 6).

The distribution of the location estimate errors for both experiments are shown in Figure 10. The errors in the first experiment are approximately normally distributed around the mean error with a few outliers at 4 m. 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.

In our previous work [Kusý et al. 2007b; Amundson et al. 2008], our system state contained five variables, the additional variable being the unknown beat frequency. Compared with our previous work, we see similar results for Experiment 1. For Experiment 2, the localization error increased, although the heading accuracy was better. A comprehensive comparison between our current and previous results is difficult due to the use of different signals; however, it is worth noting that Experiment 2 is a degenerate case and will not occur frequently. Furthermore, as discussed in Section 5, when implementing this system on mobile sensors, sudden maneuvers will *not* result in large errors because the mobile sensor is aware of any significant change in heading it makes, and provides this information directly to the Kalman filter to compensate for it.

5. NAVIGATION

In this section, we present the design of our dNav navigation system for mobile sensors that can control their own movement. Typically, robotic mobile sensors employ optical encoders, sonar, or laser rangefinders that can be used for localization and navigation. As mobile devices decrease in size and cost, these types of positioning sensors become difficult to implement. By using RF Doppler shifts as control feedback, we can design a navigation system using commercially available and inexpensive sensor nodes. Furthermore, we eliminate the need for additional hardware that may be bulky, expensive, and power-intensive.

For this system, our mobile sensor node is a two-wheeled mobile robot (WMR) with differential steering. Like the dTrack case, the mobile node is a transmitter. The receiver nodes send the observed signal frequencies back to the mobile node, and the observations are passed through an EKF in order to arrive at an estimated node trajectory in the presence of measurement noise. The output of the filter is the current estimate of the mobile node position, speed, and heading. These values are then used to calculate the trajectory error, based on a reference speed and heading setpoint. The error is passed through a controller, which outputs updated left and right wheel angular velocities.

Although we apply the same underlying methodology, there are significant fundamental differences between dTrack and dNav. Unlike dTrack, for dNav we utilize the Doppler-shifted frequency information as feedback to *control* the mobile node. This approach requires the navigation algorithm to be implemented on the mobile sensor node, within the control loop. One benefit of dNav over dTrack is that the mobile node is aware of the angular velocity commands it gives to each wheel, and this information can be used by the EKF to arrive at a better position and velocity estimate.

Figure 11 illustrates the navigation system. Each component of the architecture is presented in the following subsections.

5.1 Robot Kinematics

We use the following equations to describe the kinematic model for our WMR:

$$\dot{x} = \frac{r(\omega_r + \omega_l)}{2} cos\phi, \tag{13}$$

$$\dot{y} = \frac{r(\omega_r + \omega_l)}{2} sin\phi, \tag{14}$$

$$\dot{\phi} = \frac{r(\omega_r - \omega_l)}{2b}, \tag{15}$$

where x and y constitute the robot position, ϕ is the heading, r is the wheel radius, b is the distance between the hub center of the driving wheel and robot axis of symmetry, and ω_r and ω_l are the right and left wheel angular velocities, respectively. The speed of the robot is the magnitude of the velocity, and in terms of wheel angular velocity is represented as $|v| = \frac{r(\omega_r+\omega_l)}{2}$. For simplicity, this model does not take into account the effect of acceleration. For WMRs with

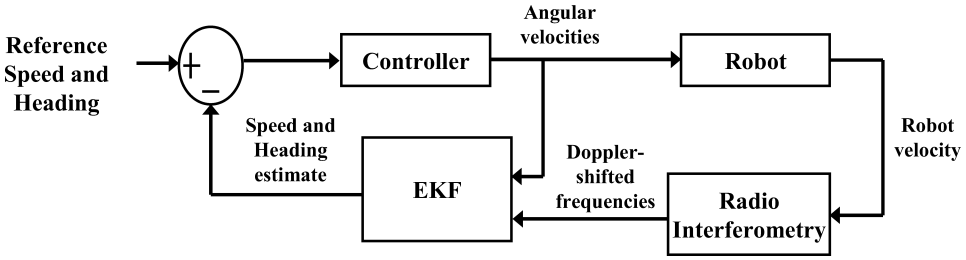

Fig. 11. The robot navigation system.

sufficiently low mass, acceleration does not have a major impact on the forward kinematics of the system.

5.2 Controller

To arrive at the angular velocities that will keep the robot on the reference trajectory, we use a controller that takes as input the speed and heading errors, $e_{|v|}$ and e_ϕ, respectively. Because ϕ wraps to 0 at 2π, we shift the heading error to fall between $-\pi$ and π:

$$e_\phi = \begin{cases} e_\phi - 2\pi, & \text{if } e_\phi > \pi, \\ e_\phi + 2\pi, & \text{if } e_\phi < -\pi, \\ e_\phi, & \text{otherwise.} \end{cases}$$

The controller contains two PI equations, one for each error component:

$$|v| = K_p e_{|v|} + K_i \int e_{|v|} dt, \tag{16}$$

$$\dot{\phi} = K_p e_\phi + K_i \int e_\phi dt, \tag{17}$$

where K_p and K_i are the proportional and integral gains, respectively. The PI equations give us the updated robot speed and angular velocity. However, the robot is driven by specifying an angular velocity for each wheel. Consequently, we convert $|v|$ and $\dot{\phi}$ into individual wheel angular velocities, ω_l and ω_r, as follows:

$$\omega_l = \frac{|v| - b\dot{\phi}}{r}, \tag{18}$$

$$\omega_r = \frac{|v| + b\dot{\phi}}{r}. \tag{19}$$

These angular velocities constitute the robot (process) input, u.

The effect of the above transformation is that both wheels will be set with an equal base velocity to compensate for the translational speed error. If heading error exists, the robot will minimize it by turning one wheel faster than the base velocity, and the other wheel slower, which will result in the robot turning in the correct direction as it moves forward.

5.3 Extended Kalman Filter

Because dNav is based on speed and heading rather than v_x and v_y velocity components, the EKF state variables are different from those in the dTrack case. For instance, the robot state is $X = \begin{bmatrix} x & y & |v| & \phi \end{bmatrix}^{\mathsf{T}}$, and the relationship between the Doppler-shifted frequency observations and the robot state (Equation (5) for the dTtrack case) becomes

$$H_i(X) = |v| cos\phi \left(cos\left(tan^{-1}\left(\frac{y_i - y}{x_i - x} \right) \right) - cos\left(tan^{-1}\left(\frac{y_{i+1} - y}{x_{i+1} - x} \right) \right) \right) - |v| sin\phi \left(sin\left(tan^{-1}\left(\frac{y_i - y}{x_i - x} \right) \right) - sin\left(tan^{-1}\left(\frac{y_{i+1} - y}{x_{i+1} - x} \right) \right) \right). \quad (20)$$

The state transition vector function F that governs the robot is given by

$$F = \begin{bmatrix} x_{k-1} + \Delta t \frac{r(\omega_r + \omega_l)}{2} cos\phi_{k-1} \\ y_{k-1} + \Delta t \frac{r(\omega_r + \omega_l)}{2} sin\phi_{k-1} \\ \frac{r(\omega_r + \omega_l)}{2} \\ \phi_{k-1} + \Delta t \frac{r(\omega_r - \omega_l)}{2b} \end{bmatrix}, \quad (21)$$

where Δt is the time elapsed since the last time step, and ω_l and ω_r comprise the process input u.

5.4 Implementation

The robot used in our experiments was a MobileRobots Pioneer 3DX,[3] with $r = 9.55$ cm and $b = 17.78$ cm. Note that, although the Pioneer is equipped with an onboard Linux PC, it was not powered on for these experiments, and all control operations were performed by the connected mote. In addition, the robot has optical encoders on each wheel; however, the measurement data was not made available to the controller at runtime. Figure 12 shows the robot and sensor nodes used in our experiments.

5.5 Evaluation

5.5.1 *Experimental Setup.* For these experiments, XSMs were used to control the robot, as well as for making the radio interferometric measurements. Eight motes were placed in a 50 by 40-m^2 sensing region, elevated 1.5 m from the ground. Seven of these were receivers, and one was the stationary assistant transmitter. Another mote, the mobile transmitter, was fixed to the robot, and communicated directly to the robot microcontroller via a serial connection. One additional mote was used to host the Kalman filter. Ideally, EKF functionality would be implemented on the same node as the controller. However, due to memory limitations, we made the design decision to use two nodes. We argue that this does not affect scalability of the system, and with code optimizations it could be possible to implement the EKF on the controller node. The EKF mote was mounted to the robot body and communicated with the controller node over the wireless radio interface.

[3]http://actirobots.com/rObOts/p2dx.html.

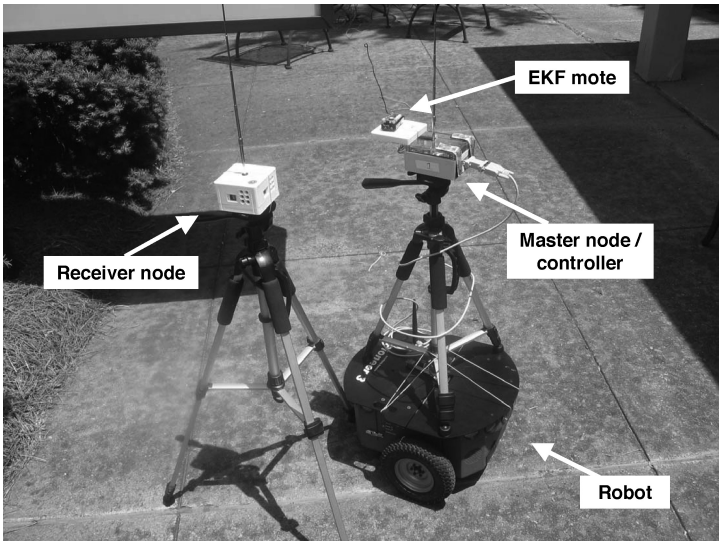

Fig. 12. Sensor hardware used in our experiments.

5.5.2 *Experimental Results.* To obtain ground truth, we used measurements from the onboard optical encoders. In addition, the sensor field was recorded by video. Because of error in the EKF position estimate, we could not rely on the robot reaching the exact waypoint coordinates. Therefore we selected an appropriate waypoint proximity region, in which the robot would consider the waypoint reached. An interesting situation arises in which the robot passes through the waypoint region between EKF updates, and is unaware it has done so. To prevent this, we had to make the waypoint region sufficiently large so that at the desired reference speed the robot would not completely pass through it. For these experiments, if the robot came within 2 m of the waypoint, it would turn and proceed to the next waypoint.

Figure 13 illustrates the desired and actual paths of the robot during waypoint navigation. With each feedback cycle, we recorded the current position, speed, and heading, and compared these with the desired values. The average position, speed, and heading error was 1.89 m, 0.19 m/s, and 12.05°, respectively.

5.6 Analysis of Experimental Results

In implementing dNav, we have made two significant contributions. First, we show that this system is moteable (i.e., can be implemented solely on mote-class devices), and that it does not require any PC processing. This is not a trivial task due to the limited memory and processing power available on the motes, as well as the intricacies of the system integration between the mote and the robot. Second, we show that our system is capable of rapid localization, which enables a mobile node to accurately navigate through a sensing region. Accurate navigation requires the latency between feedback measurements to

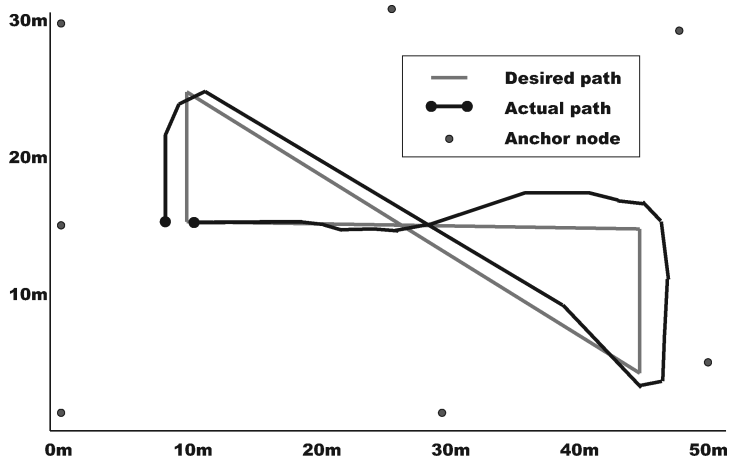

Fig. 13. Desired and actual paths of robot.

Table III. Memory Requirements and Latency for the nesC Implementation of the EKF Algorithm

	4-State	5-State
EKF Memory Usage	899 bytes	1107 bytes
EKF Latency	216 ms	265 ms

be small, which in turn requires tight synchronization between all components within the control loop.

By placing radio interferometry and the extended Kalman filter within the control loop, we are able to navigate between waypoints in a sensing region with a maximum position error of 2.51 m. Factors that contribute to this error include measurement noise (discussed in Section 4), model noise, placement of the assistant transmitter node, and execution time of the control loop. Model noise is primarily due to our approximation of the robot kinematic model, as well as inaccuracies in the process and measurement error covariances provided to the EKF. In theory, the assistant transmitter can be placed anywhere in the sensing region, as long as its signal is received at all participating nodes. However, in practice, we find that its placement relative to the mobile transmitter does in fact alter the measurement results, because the power of the assistant signal can overwhelm or be overwhelmed by the mobile transmission, which will lead to distorted signal measurements at the receiver.

The benefit of using the four-state versus the five-state solution is more apparent for our online implementation. Table III shows the memory requirements and latency of each solution implemented in nesC. Although we did not perform waypoint navigation in our previous work, and therefore cannot compare our localization results with a five-state evaluation, we realize a significant memory savings of 208 bytes by only using four states. In addition, the latency of the system decreases by 49 ms. This is important, because, in order to ensure an accurate controller response, our implementation is required to iteratively run within a small bounded timeframe. Table IV displays the average

Table IV. Execution Time of Each Component.

Component	Average (ms)	Maximum (ms)
Radio interferometry	360	361
Frequency calculation	61	116
Routing	266	580
EKF	216	224
PI update	1.3	1.4
Robot control	56	133
Total	960.3	1415.4

and maximum execution times observed over 100 feedback cycles. From these execution times, we see that we can provide feedback to the robot controller at a rate of approximately 1 Hz.

6. CONCLUSION

We have developed a novel localization algorithm for wireless sensor networks that utilizes Doppler shifts of the radio signal transmitted by a mobile node. We assumed that a number of stationary infrastructure nodes were deployed around the tracked node and that the mobile node cooperated with the localization system. We showed that Doppler shifts can be measured accurately using radio interferometry, enabling the infrastructure nodes to determine the relative speed of the tracked node with 0.13-m/s accuracy using low-cost hardware.

The localization problem was formulated as an optimization problem with the location and velocity of the mobile node being the unknown parameters, and the measured relative speeds being the constraints of the optimization. We showed that the measurement errors and the nonlinearity 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 used the constrained nonlinear least-squares (CNLS) technique to update the state of the EKF if a maneuver was detected. The combined CNLS-EKF algorithm was evaluated and achieved a location accuracy of 1.33–3.9 m in the best and the worst case, respectively. The accuracy of the speed and bearing estimates were 0.13–0.42 m/s and 7.89–16.83°, respectively.

With dTrack, only the Doppler shift measurements were computed on the sensor nodes, while the EKF-CNLS algorithm was executed on a PC. With dNav, we were able to run everything on the sensor nodes within the control loop. In this manner, we were able to successfully navigate a robot through a series of waypoints, with an average position error of 1.89 m. Although our current implementation uses separate nodes to host the controller and EKF, with code optimization it should be possible to combine the two to run on a single node.

dTrack and dNav obtain position estimates within acceptable error bounds. Naturally, these bounds are based on a set of assumptions and conditions about the system. Because we have four unknown state variables, we require at least

five receivers to make four frequency difference observations. However, this may in itself not be enough to provide a location estimate. For example, if all five receiver nodes are nonidentical collinear and the mobile node and its velocity vector lie on this line, the system has infinitely many solutions. If the mobile node is not on the line, then the system has at least two mirror solutions on the two half planes defined by the line. Thus, a necessary condition is noncollinearity of the infrastructure nodes. We assume our sensing region contains the appropriate density of infrastructure nodes to support this condition. We also assume that the two transmitters that comprise the interference signal are appropriately separated at all times so as not to overwhelm the complementary signal. In addition, implementation parameters add extra constraints. For example, we can measure Doppler shifts only up to a certain maximum speed, due to the limited sampling frequency of the RSSI signal.

The sensing region can have a virtually unlimited number of infrastructure nodes because they act as signal receivers, so there is no transmission contention among them. However, the number of mobile transmitters will be limited. If multiple mobile nodes are traversing the sensing region simultaneously, the system is still feasible, as long as a transmission schedule is agreed upon, giving each mobile node a specified transmission timeslot. Scalability of the system is then limited by the number of transmissions that can be made before a mobile node must make its next transmission. An alternative approach would be to have different mobile nodes transmit at different frequencies, and assign each frequency a sufficient number of receivers. The sensing region would then require a specific node density to insure that each transmission is observed by at least five non-collinear receivers.

Our approach is less susceptible to multipath propagation than the original 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.

ACKNOWLEDGMENTS

The authors would like to thank Gyorgy Balogh, Manish Kushwaha, and Ryan Thibodeaux for their assistance with this work, as well as Metropolitan Nashville Parks and Recreation and Edwin Warner Park for use of their space.

REFERENCES

AMUNDSON, I., KOUTSOUKOS, X., AND SALLAI, J. 2008. Mobile sensor localization and navigation using RF doppler shifts. In *Proceedings of the 1st ACM International Workshop on Mobile Entity Localization and Tracking in GPS-less Environments (MELT)*.

AMUNDSON, I., KUSHWAHA, M., AND KOUTSOUKOS, X. 2009. On the feasibility of determining angular separation in mobile wireless sensor networks. In *Proceedings of the 2nd International Workshop on Mobile Entity Localization and Tracking in GPS-less Environments (MELT)*. Lecture Notes in Computer Science, vol. 5801, Springer, Berlin, Germany.

BAHL, P. AND PADMANABHAN, V. N. 2000. Radar: An in-building RF-based user-location and tracking system. In *Proceedings of IEEE INFOCOM.* vol. 2, 77584.

BERGBREITER, S. AND PISTER, K. S. J. 2003. CotsBots: An off-the-shelf platform for distributed robotics. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

BROOKS, R. R., GRIFFIN, C., AND FRIEDLANDER, D. S. 2002. Self-organized distributed sensor network entity tracking. *Int. J. High Perform. Comput. Appl. 16,* 3.

BROOKS, R. R., RAMANTHAN, P., AND SAYEEED, A. 2003. Distributed target classification and tracking in sensor networks. *Proc. IEEE 91*, 8, 1163–1171.

BUTLER, Z., CORKE, P., PETERSON, R., AND D.RUS. 2004. Networked cows: Virtual fences for controlling cows. In *Proceedings of the MobiSys Workshop on Applications of Mobile Embedded Systems. (WAMES)*.

CHAN, Y. AND TOWERS, J. 1992. Passive localization from doppler-shifted frequency measurements. *IEEE Trans. Signal Process. 40,* 10. 2594–2598.

CHAN, Y.-T. AND JARDINE, F. 1990. Target localization and tracking from doppler-shift measurements. *IEEE J. Oceanic Eng. 15,* 3, 251–257.

CHANG, C.-B. AND TABACZYNSKI, J. 1984. Application of state estimation to target tracking. *IEEE Trans. Aut. Contr. 29–2*, 98–109.

CHANG, H.-L., TIAN, J.-B., LAI, T.-T., CHU, H.-H., AND HUANG, P. 2008. Spinning beacons for precise indoor localization. In *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems (SenSys)*.

DANTU, K., RAHIMI, M., SHAH, H., BABEL, S., DHARIWAL, A., AND SUKHATME, G. S. 2005. robomote: Enabling mobility in sensor networks. In *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN)*.

DUTTA, P., GRIMMER, M., ARORA, A., BIBYK, S., AND CULLER, D. 2005. Design of a wireless sensor network platform for detecting rare, random, and ephemeral events. *In Proceedings of IPSN/SPOTS*.

ESTRIN, D., GOVINDAN, R., HEIDEMANN, J., AND KUMAR, S. 1999. Next century challenges: Scalable coordination in sensor networks. *In Proceedings of MobiCom*.

FANG, L., ANTSAKLIS, P. J., MONTESTRUQUE, L., MCMICKELL, M. B., LEMMON, M., SUN, Y., FANG, H., KOUTROULIS, I., HAENGGI, M., XIE, M., AND XIE, X. 2005. Design of a wireless assisted pedestrian dead reckoning system: The NavMote experience. *IEEE Trans. Instrument. Measurem. 54,* 6, 2342–2358.

FRIEDMAN, J., LEE, D. C., TSIGKOGIANNIS, I., WONG, S., CHAO, D., LEVIN, D., KAISERA, W. J., AND SRIVASTAVA, M. B. 2005. Ragobot: A new platform for wireless mobile sensor networks. In *Proceedings of the International Conference on Distributed Computing in Sensor Systems (DCOSS)*.

GANDHAM, S., DAWANDE, M., PRAKASH, R., AND VENKATESAN, S. 2003. Energy efficient schemes for wireless sensor networks with multiple mobile base stations. In *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM)*.

GIROD, L., LUKAC, M., TRIFA, V., AND ESTRIN, D. 2006. The design and implementation of a self-calibrating acoustic sensing platform. In *Proceedings of ACM SenSys*.

GU, L., JIA, D., VICAIRE, P., YAN, T., LUO, L., TIRUMALA, A., CAO, Q., STANKOVIC, J. A., ABDELZAHER, T., AND KROGH, B. 2005. Lightweight detection and classification for wireless sensor networks in realistic environments. In *Proceedings of ACM SenSys*.

GUSTAFSSON, F. AND GUNNARSSON, F. 2005. Mobile positioning using wireless networks. *IEEE Signal Process. Mag. 22,* 4, 41–53.

WENER-ALLEN, G., JOHNSON, J., RUIZ, M., LEES, J., AND WELSH, M. 2005. Monitoring volcanic eruptions with a wireless sensor networks. In *Proceedings of the 2nd European Workshop on Wireless Sensor Networks. (EWSN)*.

HAZAS, M., SCOTT, J., AND KRUMM, J. 2004. Location-aware computing comes of age. *IEEE Comput. 37,* 2, 95–97.

HIGHTOWER, J. AND BORRIELLO, G. 2001. Location systems for ubiquitous computing. *IEEE Comput. 34,* 8, 57–66.

HILL, J. AND CULLER, D. 2002. Mica: a wireless Platform for deeply embedded networks. *IEEE Micro 22,* 6, 12–24.

HILL, J., SZEWCZYK, R., WOO, A., HOLLAR, S., CULLER, D., AND PISTER, K. 2000. System architecture directions for networked sensors. In *Proceedings of ASPLOS-IX*.

HU, L. AND EVANS, D. 2004. Localization for mobile sensor networks. In *Proceedings of MobiCom*.

JUANG, P., OKI, H., WANG, Y., MARTONOSI, M., PEH, L., AND RUBENSTEIN, D. 2002. Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet. In *Proceedings of ASPLOS-X*.

KALMAN, R. E. 1960. A new approach to linear filtering and prediction problems. *Trans. ASME—J. Basic Eng. 82* (Series D), 35–45.

KUSÝ, B., BALOGH, G., LÉDECZI, A., SALLAI, J., AND MARÓTI, M. 2007a. inTrack: High precision tracking of mobile sensor nodes. In *Proceedings of the 4th European Workshop on Wireless Sensor Network. (EWSN)*.

KUSÝ, B., BALOGH, G., VÖLGYESI, P., SALLAI, J., NÁDAS, A., LÉDECZI, A., MARÓTI, M., AND MEERTENS, L. 2006. Node-density independent localization. In *Proceedings of IPSN/SPOTS*.

KUSÝ, B., DUTTA, P., LEVIS, P., MARÓTI, M., LÉDECZI, A., AND CULLER, D. 2006b. Elapsed time on arrival: a simple and versatile primitive for canonical time synchronization services. *Int. J. Ad Hoc Ubiq. Comput. 2,* 1.

KUSÝ, B., LÉDECZI, A., AND KOUTSOUKOS, X. 2007b. Tracking mobile nodes using RF doppler shifts. In *Proceedings of the 5th International Conference on Embedded Networked Sensor Systems (SenSys)*. ACM, New York, NY, 29–42.

KUSÝ, B., SALLAI, J., BALOGH, G., LÉDECZI, A., PROTOPOPESCU, V., TOLLIVER, J., DENAP, F., AND PARANG, M. 2007c. Radio interferometric tracking of mobile wireless nodes. In *Proceedings of MobiSys*.

LEDECZI, A., VOLGYESI, P., SALLAI, J., AND THIBODEAUX, R. 2008. A novel RF ranging method. In *Proceedings of the 6th Workshop on Intelligent Solutions in Embedded Systems (WISES)*.

LEVIS, P., MADDEN, S., POLASTRE, J., SZEWCZYK, R., WHITEHOUSE, K., WOO, A., GAY, D., HILL, J., WELSH, M., BREWER, E., AND CULLER, D. 2005. Tinyos: An operating system for wireless sensor networks. *Ambient Intelligence*. Sprinter-Verlag, Berlin, Germany.

LI, X. R. AND JILKOV, V. P. 2004. A survey of maneuvering target tracking: approximation techniques for nonlinear filtering. In *Proceedings of the SPIE Conference on Signal and Data Processing of Small Targets*. 537–550.

MADSEN, K., NIELSEN, H., AND TINGLEFF, O. 2004. Methods for nonlinear least squares problems. Lecture Note. Technical University of Denmark Lyngby, Denmark.

MARÓTI, M., KUSÝ, B., BALOGH, G., VÖLGYESI, P., NÁDAS, A., MOLNÁR, K., DÓRA, S., AND LÉDECZI, A. 2005. Radio interferometric geolocation. In *Proceedings of ACM SenSys*.

MOORE, D., LEONARD, J., RUS, D., AND TELLER, S. 2004. Robust distributed network localization with noisy range measurements. In *Proceedings of ACM Sensys*.

SHAH, R., ROY, S., JAIN, S., AND BRUNETTE, W. 2003. Data mules: Modeling a three-tier architecture for sparse sensor networks. In *Proceedings of the 1st IEEE International Workshop on Sensor Network Protocols and Applications*.

SMITH, A., BALAKRISHNAN, H., GORACZKO, M., AND PRIYANTHA, N. 2004. Tracking moving devices with the Cricket location system. In *Proceedings of MobiSys*.

TAYLOR, C., RAHIMI, A., BACHRACH, J., SHROBE, H., AND GRUE, A. 2006. Simultaneous localization, calibration, and tracking in an ad hoc sensor network. In *Proceedings of IPSN*. 27–33.

WANG, G., CAO, G., PORTA, T., AND ZHANG, W. 2005. Sensor relocation in mobile sensor networks. In *Proceedings of the IEEE INFOCOM*.

WELCH, G. AND BISHOP, G. 2004. An introduction to the Kalman filter. Tech. rep. TR 95-041. Department of Computer Science, University of North Carolina at Chapel Hill, Chapel Hill, NC.

XU, N., RANGWALA, S., CHINTALAPUDI, K. K., GANESAN, D., BROAD, A., GOVINDAN, R., AND ESTRIN, D. 2004. A wireless sensor network for structural monitoring. In *Proceedings of ACM SenSys*.

ZHAO, F., LIU, J., LIU, J., GUIBAS, L., AND REICH, J. 2003. Collaborative signal and information processing: An information directed approach. In *Proceedings of IEEE 91,* 8, 1199–1209.

Received March 2009; revised November 2009; accepted November 2009