Node Localization Using Mobile Robots in Delay-Tolerant Sensor Networks

Pubudu N Pathirana† Andrey V Savkin‡ Sanjay Jha§ Nirupama Bulusu ¶

Abstract

We present a novel scheme for node localization in a Delay-Tolerant Sensor Network (DTN). In a DTN, sensor devices are often organized in network clusters that may be mutually disconnected. Some mobile-robots may be used for data collection from the network clusters. The key idea in our scheme is to use

---

†This work is supported by the Australian Research Council and National ICT Australia Limited.
‡School of Engineering and Technology, Deakin University, Victoria 3217, Australia. Tel : +(61 3) 52272914, Fax : +(61 3) 52272167, E-mail pubudu@deakin.edu.au
§School of Electrical Engineering and Telecommunications, University of New South Wales, Sydney 2052, Australia, Tel : +(61 2) 93856359, Fax : +(61 2)93855993, E-mail : a.savkin@unsw.edu.au
¶School of Computer Science and Engineering, University of New South Wales, Sydney 2052, Australia. Tel : +(61 2) 93856471, Fax : +(61 2) 9385 5770, E-mail : sjha@cse.unsw.edu.au

*Corresponding Author.

§National ICT Australia Limited, Bay 15, Australian Technology Park, Eveleigh 1431, Australia. Tel : +(61 2) 8374 5507, E-mail : nbulusu@cse.unsw.edu.au
this robot to perform location estimation for the sensor nodes it passes by based on the signal strength of radio messages received from them. Thus we eliminate the processing constraints of static sensor nodes and the need for static reference beacons. Our mathematical contribution is the use of a Robust Extended Kalman Filter (REKF) based state estimator to solve the localization. Compared to the standard extended Kalman filter, REKF is computationally efficient and also more robust, since it does not make any assumptions about the measurement noise. Finally, we have implemented our localization scheme on a hybrid sensor network testbed, and show that it can achieve node localization accuracy within 1m in a large indoor setting. **Index Terms**: localization, delay-tolerant sensor networks, Robust Extended Kalman Filter, mobile robot, mobility.

1 Introduction

Recent years have witnessed a boom in sensor networks research [3, 9] and commercial activities[7]. This has been motivated by the wide range of potential applications from environmental monitoring to condition-based maintenance of aircraft. Sensor networks are frequently envisioned to exist at large scale, and characterized by extremely limited end-node power, memory and processing capability.
The concept of a *delay-tolerant sensor network (DTN)* was first proposed by Fall[12]. A DTN would typically be deployed to monitor an environment over a long period of time, and characterized by non-interactive sensor data traffic. Sensors are randomly scattered and organize into one or more clusters that may be disconnected from each other. Each cluster has a cluster-head. Sensor information is typically aggregated at the cluster heads, which tend to have more resources and are responsible for communicating data to outside world. Wireless mobile robots (e.g. robomote [25]), unmanned aerial vehicles can roam around the network to collect data from cluster heads, or to dynamically reprogram or reconfigure the sensors. Examples of DTNs in existence are Sammi[16], Zebranet[11], and DataMules[24].

This paper revisits the problem of *node localization*, i.e., estimating sensor node positions for a delay-tolerant sensor network. A DTN has several distinguishing characteristics which motivate alternate approaches to node localization than those previously proposed. In a DTN, sensor nodes need neither be localized in real time, nor all at once. In this paper, we propose a novel localization scheme for DTNs using received signal strength (RSSI) measurements from each sensor device at a data gathering mobile-robot. Our contributions are threefold:

- We motivate and propose a novel approach that allows one/more mobile robots to perform node localization in a DTN, eliminating the processing constraints
of small devices. Mobility can also be exploited to reduce localization errors and the number of static reference location beacons required to uniquely localize a sensor network.

- We develop a novel Robust Extended Kalman Filter (REKF) [18], based state estimation algorithm for node localization in DTNs. Localization based on signal strength measurements is solved by treating it as on-line estimation in a nonlinear dynamic system (Section 3). Our model incorporates significant uncertainty and measurement errors and is computationally more efficient and robust in comparison to the extended Kalman filter implementation used to solve similar problem in cellular networks [14] [5] (simulations, section 4).

- We implement and validate our scheme on a novel hybrid sensor network testbed of motes, Stargates and Lego Mindstorm robots (section 5).

2 Related Work

In this section, we review research most relevant to our work — (i) delay-tolerant sensor networks, and (ii) sensor network localization.
2.1 Delay-tolerant Sensor Networks

Fall first proposed a Delay-tolerant Network architecture[12] for sensors deployed in mobile and extreme environments lacking an always-on infrastructure. These sensors are envisioned to monitor the environment over a long period of time. Herein, communication is based on an abstraction of message switching rather than packet switching. The abstraction of moderate-length message (known as bundles) delivery for non-interactive traffic can provide benefits for network management because it allows the network path selection and scheduling functions to have a-priori knowledge about the size and performance requirements of requested data transfers.

DTNs are already being used in practice. DataMules[24] uses a Mule that periodically visits sensor devices and collects information from these devices, in effect providing a message store-and-forward service, enabling low-power sensor nodes to conserve power. The Sammi Network[16] is a community of Sammi people, who are reindeer herders in Sweden who keep relocating their base. The Sammi communities do not have a wired or wireless communication infrastructure. Their relocation is controlled by an yearly cycle which depends on the natural behaviour of reindeer. In the Zebranet wildlife tracking system [11], wireless sensor nodes attached to animals collect location data and opportunistically report their histories when they come within radio range of base stations. While previous research has focused on
communication abstractions, we are investigating the challenges and opportunities that arise from mobile data collecting elements in DTNs.

2.2 Sensor Network Localization

Localization is one of the most widely researched topics within the area of sensor networks, and in robotics [2, 19, 23, 6]. Previous localization systems for sensor networks [19, 23] have been designed to simultaneously scale and continuously localize a large number of devices. To meet these requirements, devices usually compute their own location from distance (or other measurements) made to nearby reference beacons. However, localization requirements for these sensor networks are different from delay-tolerant sensor networks.

In a DTN, nodes neither need to be localized concurrently nor continuously. We tradeoff computational time for node localization for several other benefits. We can reduce the computational requirements for small sensor devices, by instead using one or more mobile robots to compute the location of sensors. We can employ more sophisticated algorithms since processing is performed by the robot rather than sensor devices. We can also reduce the number of static location reference beacons required, by exploiting the mobility of the robot.  

\footnote{For instance, Eren et al [4] estimate that to uniquely localize a sensor network of \( n \) nodes in \( O(\sqrt{\log n}) \) steps, \( O(\frac{1}{\log n}) \) reference location beacons are required using the iterative trilateration scheme proposed in [23]. To localize with just 1 beacon, \( O(n) \) steps will be required.}
Previous research has also investigated RSSI-based localization schemes[1]. One of the main drawbacks in RSSI-based localization schemes is the RSSI measurement noise caused by short-scale and medium-scale fading. In our scheme, we reduce the impact of fading, to make RSSI-based localization more viable. Because the robot-receiver is mobile, over a period of time, we can statistically eliminate the fading noise in RSSI measurements (not possible with static transmitter-receiver pair).

Previously, Kalman filters and Bayesian filters have been applied to the localization problem (mainly in the context of robotics and cellular networks). In this paper we propose using a Robust Extended Kalman Filter (REKF) as a state estimator in predicting sensor locations. These robust state estimation ideas emerged from the work of Savkin and Petersen[22]. It not only provides satisfactory results[17], but also eliminates the requirement of the knowledge of measurement noise in the more commonly used standard Kalman filter implementation presented in [14]. It is significantly more computation and memory efficient than the more adaptive, but computationally complex and memory-intensive Bayesian filters; making it better suited to the sensor networks regime. In the next section, we describe in detail, our node localization scheme using mobile robots in a delay-tolerant sensor network.

---

In our scheme, assume that the mobile robot can localize $O(\log n)$ sensors in each step (a single Filter computation). This is not an unreasonable assumption to make since $\log n \leq 10$, even for very large $n$. Using just 1 mobile robot, node localization can be achieved in $O\left(\frac{n}{\log n}\right)$ steps. To localize in $O(\sqrt{\log n})$ steps, our scheme requires $O\left(\frac{n}{\log n \sqrt{\log n}}\right)$ steps.
3 Localization Methodology

To solve node localization based on RSSI measurements at a mobile-robot, we model it as an on-line estimation in a nonlinear dynamic system. In this section, we describe this system dynamic model and the nonlinear measurement model. We present the theoretical background for the Robust Extended Kalman filter used with this model in Appendix A.

3.1 System dynamic model

We use the terminology mobile – robot for a mobile-node fitted with a wireless base-station. The sensors to be located are randomly distributed in an environment. The dynamic model for n sensors and the mobile-robot can be given in two dimensional cartesian coordinates as [21]

\[ \dot{x}(t) = Ax(t) + B_1 u(t) + B_2 w(t) \]  

(3.1)

where

\[ A = \begin{bmatrix} \Theta & 0 \\ \vdots & \ddots \\ 0 & \Theta \end{bmatrix}, \quad -B_1 = \begin{bmatrix} \Phi \\ \vdots \\ \Phi \end{bmatrix} \]
The dynamic state vector \( x(t) = [x_1(t) \ldots x_i(t) \ldots x_n(t)]' \) with \( x_i(t) = [\text{X}_i(t) \ \text{Y}_i(t) \ \dot{\text{X}}_i(t) \ \dot{\text{Y}}_i(t)]' \), where \( i \in [1 \ldots n] \) and \( \text{X}_i(t) \) and \( \text{Y}_i(t) \) represent the position of the \( i^{th} \) sensor (Sensor\(_i\)) with respect to the mobile–robot at time \( t \), and their first order derivatives \( \dot{\text{X}}_i(t) \) and \( \dot{\text{Y}}_i(t) \) represent the relative speed along the X and Y directions. In other words, if \( x_c(t) = [x_c(t) \ y_c(t) \ \dot{x}_c(t) \ \dot{y}_c(t)]' \) represent the absolute state (position and velocity in order in the X and Y direction respectively) of the (mobile–robot) and \( x_s^i(t) = [\text{X}_s^i(t) \ \text{Y}_s^i(t) \ \dot{\text{X}}_s^i(t) \ \dot{\text{Y}}_s^i(t)]' \) denote the absolute state of the Sensor\(_i\) in the same order, then \( x_i(t) \triangleq x_c(t) - x_s^i(t) \). Furthermore, let \( u(t) \) denote the two dimensional driving/acceleration command of the mobile–robot from the respective accelerometer readings and \( w(t) \) denote the unknown two-dimensional driving/acceleration command of the sensor if moving. Although it can be generalized for moving sensor
case, as most applications rely on stationary sensors, here we consider the sensors as stationary and set $w(t) = 0$. This system can be represented in graphical form in the form of an input ($u(t)$) and measurement ($y$) system as in Figure 3.1. We omitted $B_2$ as we only consider the case of stationary sensors. The basic idea in such a system is to estimate state $x$ from measurement $y$. In the localization problem, as the sensor locations are unknown, we assume an arbitrary location $(0,0)$. We show that this assumed state converges to the actual state and hence the unknown sensor location can be estimated (as the position/state of the mobile – robot is known) within the prescribed time frame.

![Figure 3.1: Location estimation system](image)

### 3.2 RSSI Measurement model

In wireless networks, the distance between two communicating entities is observable using the forward link RSSI (received signal strength indication) of the receiver. Measured in decibels at the mobile-robot for our case, RSSI can be modelled as
a two folds effect: due to path loss and due to shadow fading[14]. Fast fading is
neglected assuming that a low-pass filter is used to attenuate Rayleigh or Rician
fade. Denoting the $i^{th}$ sensor as Sensor$_i$ (figure 3.2), the RSSI from the Sensor$_i$, $p_i(t)$

$$p_i(t) = p_{oi} - 10\varepsilon \log d_i(t) + v_i(t),$$  \hspace{1cm} (3.2)

where $p_{oi}$ is a constant determined by transmitted power, wavelength, and antenna
gain of the mobile – robot. $\varepsilon$ is a slope index (typically 2 for highways and 4 for
microcells in the city), and $v_i(t)$ is the logarithm of the shadowing component, which
is considered as an uncertainty in the measurement. $d_i(t)$ represents the distance be-
tween the mobile – robot and Sensor$_i$, which can be further expressed in terms of the

Figure 3.2: Network Geometry
position of the \( i^{th} \) sensor with respect to the location mobile-robot i.e., \((x_i(t), y_i(t))\)

\[
d_i(t) = \left( x_i(t)^2 + y_i(t)^2 \right)^{1/2}
\]  

(3.3)

For sensors within a network cluster, we uses measurements at a single mobile-robot as opposed to multiple ones[14]. The observation vector

\[
y(t) = \begin{bmatrix} p_1(t) \\ \vdots \\ p_n(t) \end{bmatrix},
\]  

(3.4)

is sampled progressively as the mobile – robot moves in the coverage area. The measurement equation for the measurements made by the mobile – robot for the \( n \) number of sensors are in the form of,

\[
y(t) = C(x(t)) + v(t)
\]  

(3.5)
where \( v(t) = [v_1(t) \cdots v_n(t)]' \) with
\[
C(x(t)) = \begin{bmatrix}
    p_{o1} - 10\varepsilon \log (x_1(t)^2 + y_1(t)^2) \\
    \vdots \\
    p_{on} - 10\varepsilon \log (x_n(t)^2 + y_n(t)^2)
\end{bmatrix}.
\]

We provide a brief intuitive explanation of REKF here (see Appendix for detailed theoretical background). We use the state space model (a representation of the dynamic system consisting of the mobile – robot and the \( n \) sensors using a set of differential equations derived from simple kinematic equations). Our dynamic system considers two noise inputs: (i) measurement noise (this is standard with any measurement), \( v \) in \( y = C(x) + v \) and (ii) \( w \) - acceleration is also considered noise as it is unknown. In this application, the initial condition errors are quite significant as no knowledge is available regarding the sensor locations. This issue is directly addressed as the proposing algorithm is inherently robust against estimation errors of the initial condition (see equation A.6 in the appendix). The two noise inputs and the initial estimation errors have to satisfy IQC equation (equation A.6 in the appendix). If there exists a solution to the Ricati equation (see section A.2 in the appendix), then the IQC presented in a suitable form by the equation A.6 is satisfied and the states can be estimated from the measurements using equation A.8 which is
a Robust version of the Extended Kalman Filter (REKF).

In the application of REKF in a delay-tolerant network, the $i^{th}$ system (mobile — robot and the Sensor$_i$) during a corresponding time interval is represented by the nonlinear uncertain system in (A.1) together with the following Integral Quadratic Constraint (IQC) (from equation A.6):

$$(x(0) - x_0)' N_i (x(0) - x_0) + \frac{1}{2} \int_0^s \left( w(t)' Q_i(t) w(t) \right) + v(t)' R_i(t) v(t) dt \\ \leq d + \frac{1}{2} \int_0^s z(t)' z(t) dt.$$  \hspace{1cm} (3.7)

Here $Q_i > 0, R_i > 0$ and $N_i > 0$ with $i \in \{1, 2, 3\}$ are the weighting matrices for each system $i$. The initial state($x_0$) is the estimated state of respective systems at startup. It is essentially derived from the terminal state of the previous system together with other data available in the network (i.e. robot position and speed) to be used as the initial state for the next system taking over the tracking. With an uncertainty relationship of the form of (3.7), the inherent measurement noise (see equation 3.5), unknown mobile robot acceleration and the uncertainty in the initial condition are considered as bounded deterministic uncertain inputs. In particular, the measurement equation with the standard norm bounded uncertainty can be written as
(equation 3.5)

\[ y = C(x) + \delta C(x) + v_0 \]  

(3.8)

where \(|\delta| \leq \xi\) with \(\xi\), a constant indicating the upper bound of the norm bounded portion of the noise. By choosing \(z = \xi C(x)\) and \(\nu = \delta C(x)\),

\[ \int_0^T |\nu| dt \leq \int_0^T z' z dt. \]  

(3.9)

Considering \(v_0\) and the corresponding uncertainty in \(w\) as \(w_0\) satisfying the bound

\[ \Phi(x(0)) + \int_0^T [w_0(t)'Qw_0(t) + v_0(t)'Rv_0(t)] dt \leq d, \]  

(3.10)

it is clear that this uncertain system leads to the satisfaction of condition in inequality A.2 and hence A.6 (see [18]). This more realistic approach removes any noise model assumptions in algorithm development and guarantees the robustness.

### 3.3 Robust versus Optimal State Estimation

REKF tends to increase the robustness of the state estimation process and reduce the chance that a small deviation from the Gaussian process in the system noise causes a significant negative impact on the solution. However, we lose optimality
and our solution will be just sub-optimal. To explain the connection between REKF and the standard extended Kalman Filter, consider the system A.1 with

\[ K(x, u) = \nu K_0(x, u) \]  \hspace{1cm} (3.11)

where \( K_0(x, u) \) is some bounded function, and \( \nu > 0 \) is a parameter. Then, the REKF estimate \( \hat{x}(t) \) for the system A.1, 3.11, A.6 defined by A.8, A.9 converges to \( \hat{x}^0(t) \) as \( \nu \) tends to 0. Here \( \hat{x}^0(t) \) is the extended Kalman state estimate for the system A.1 with the Gaussian noise \( \begin{bmatrix} w(t)' \quad v(t)' \end{bmatrix} \) satisfying

\[ E \left\{ \begin{bmatrix} w(t) \\ v(t) \end{bmatrix} \begin{bmatrix} w(t)' \\ v(t)' \end{bmatrix} \right\} = \begin{bmatrix} Q(t) & 0 \\ 0 & R(t) \end{bmatrix} ; \]

The parameter \( \nu \) in 3.11 describes the uncertainty in the system and measurement noise. For small \( \nu \), our robust state estimate becomes close to the Kalman state estimate with Gaussian noise. For larger \( \nu \), we achieve more robustness but less optimality. We show via simulation that for larger uncertainty (which is quite realistic) our robust filter still performs well whereas the standard extended Kalman estimate diverges.
4 Simulations

To examine the performance of the Robust Extended Kalman Filter for a sensor network, we simulate a mobile-robot equipped with a radio transceiver moving in the sensor coverage area. We assume the network knows the acceleration of the mobile-robot via GPS and accelerometer readings but has no information about the sensors. We simulate two scenarios — large sensors and small sensors.

4.1 Large Sensors

In Scenario I, we simulate large sensors scattered over a wide area. We expect sensors to be low cost and equipped with modest transmitters. To model this, we use a slow sampling rate of 2 sec per sample. We simulate a mobile robot and four sensors. The algorithm can be scaled to as many sensors as required by appropriately increasing the number of mobile robots. Simulation parameters are listed in Table 4.1. The mobile-robot measures the forward link signal from four sensors and estimates the state of the system from an arbitrary initial estimate (zero). Figure 4.1 shows how the estimated sensor location from an initial position converges to the actual sensor positions within the simulation time. Figure 4.2 shows the distance variation in X and Y directions separately for each sensor as well as the predicted distances approaching the actual distances. Figure 4.4 shows that the extended Kalman filter
cannot be used with large uncertain instances as it diverges.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_{os}$</td>
<td>20W</td>
<td>Base station transmission power</td>
</tr>
<tr>
<td>$N$</td>
<td>diag${1,1,1,1,70,25,2,5,100,1,20,1,40,40,20,1}$</td>
<td>Weighting on the Initial viscosity solution</td>
</tr>
<tr>
<td>$Q$</td>
<td>diag${2,2,11,1,1,3,1,2}$</td>
<td>Weighting on the uncertainty in the vehicle driving command</td>
</tr>
<tr>
<td>$R$</td>
<td>diag${2 \times 10^4, 1.7 \times 10^3, 10 \times 10^3, 51 \times 10^3}$</td>
<td>Weighting on the measurement noise</td>
</tr>
<tr>
<td>$T$</td>
<td>5 mins</td>
<td>Simulation time</td>
</tr>
<tr>
<td>$A_{max}$</td>
<td>50 m/s$^{-2}$</td>
<td>weighting on $u(t)$</td>
</tr>
<tr>
<td>$T_s$</td>
<td>2 s</td>
<td>Sampling interval</td>
</tr>
<tr>
<td>$x_1^1(0)$</td>
<td>$[500\text{m} \ 2500\text{m} \ 0 \ 0]'$</td>
<td>1st sensor initial state</td>
</tr>
<tr>
<td>$x_2^2(0)$</td>
<td>$[2500\text{m} \ 500\text{m} \ 0 \ 0]'$</td>
<td>2nd sensor initial state</td>
</tr>
<tr>
<td>$x_3^3(0)$</td>
<td>$[1 \ 600\text{m} \ 900\text{m} \ 0 \ 0]'$</td>
<td>3rd sensor initial state</td>
</tr>
<tr>
<td>$x_4^4(0)$</td>
<td>$[2500\text{m} \ 1250\text{m} \ 0 \ 0]'$</td>
<td>4th sensor initial state</td>
</tr>
<tr>
<td>$x_c(0)$</td>
<td>$[2500\text{m} \ -2500\text{m} \ 20\text{ms}^{-1} \ 10\text{ms}^{-1}]'$</td>
<td>mobile - robot initial state</td>
</tr>
</tbody>
</table>

Table 4.1: Simulation parameters for Scenario I.

4.2 Small Sensors

In Scenario II, we use sensors with much lesser signal strength (600mW) as in [10] with higher sampling rate as in commercially available systems. To demonstrate scalability, we increase both the time scale and the number of sensors in this scenario. The simulation parameters are given in Table 7.1. The arbitrary acceleration of the mobile robot is taken as $u(t) = A_{max} [-3 \sin(0.2t) + \phi_1 \ 0.9 \cos(0.05t) + \phi_2]$. 

18
Figure 4.1: Location estimation trajectories converging to the actual sensor locations.

In the dynamical system simulation, we choose the functions given in Table 7.1 for the arbitrary mobile-robot acceleration \( (u) \), with \( \phi_1 \), and \( \phi_2 \) being uniform random distributions in the interval \([0 \ 0.1A_{\text{max}}]\). We consider uniformly distributed measurement noise in the interval \([0 \ 0.01\|y(t)\|]\) with \(y(t)\) being the noise free measurement with \(\xi = 0.05\). The equation for the state estimation and the corresponding Riccati Differential equation obtained from equation A.8 and A.9 are:

\[
\dot{x}(t) = A\hat{x}(t) + B_1u_i(t)
\]
Figure 4.2: Estimation convergence for sensors 1, 2, 3 and 4.

\[ +X^{-1}(t)\beta_1(\dot{x}(t))' R_{ij}(y(t) - \beta(\ddot{x}(t))) + \xi \beta_1(\dot{x}(t))' \beta_1(\dot{x}(t)) = 0 \]

\[ \ddot{x}(t) = x_0, \]  

\[ \dot{X} + A'X + AX + X B_2 Q_i^{-1} B_2' X - \beta \dot{x}(t)' R_{ij} \beta_1 \dot{x}(t) + \xi \beta_1(\dot{x}(t))' \beta_1(\dot{x}(t)) = 0 \]

\[ X(0) = N, \]

where

\[ \beta(x) = C(x(t)) \]

as shown in equation 3.6. Also here

\[ \beta_1(x) = \nabla_x \beta(x), \]
Figure 4.3: (a) Estimation path for each sensor (Scenario 2). (b) Percentage error and the absolute error from the initial estimation.

$x_0$ is $x(0)$, the relative initial dynamic state of the system. In the second scenario, ten sensors with lesser signal strength are used with the mobile robot. Figure 4.3(a) plots the estimated trajectory approaching each respective sensor location from an initial estimate of each sensor location of (0,0). Figure 4.3(b) plots the percentage error in localizing each sensor with respect to the initial estimation error.

5 Implementation and Experimental Results

We have implemented our REKF-based localization scheme to verify its computational efficiency and estimation accuracy in a real environment. We now describe this implementation and report on preliminary experimental results.
Figure 4.4: (a) Using standard Kalman filter as the state estimator. (b) Divergence of state estimation when using standard Kalman filter.

5.1 Hybrid Sensor Platform

Our hybrid platform consists of three devices — (i) motes (ii) Stargates and (iii) Lego Mindstorm robots. They differ in processing, memory, battery, and mobility capabilities; and also in their operating systems software.

5.1.1 Motes

Motes[15], shown in Figure 5.1, are our resource impoverished devices and run the TinyOS event-driven operating system. The Mica2 mote sensors deployed in our experiment use the CC1000 radio from ChipCon, which provides an analog RSSI measurement that can be connected to an analog to digital converter (ADC) to
<table>
<thead>
<tr>
<th>Platform</th>
<th>Core</th>
<th>Processor</th>
<th>Data Path</th>
<th>MHz</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>MICA2</td>
<td>Atmega 128L4</td>
<td>RISC</td>
<td>8 bits</td>
<td>4 MHz</td>
<td>128 KB program flash, 4 KB data</td>
</tr>
<tr>
<td>Stargate</td>
<td>Intel PXA255 XScale</td>
<td>RISC</td>
<td>32 bits</td>
<td>400 MHz</td>
<td>32 MB flash, 64 MB SDRAM</td>
</tr>
</tbody>
</table>

Table 5.1: Stargate and MICA2 Comparison

produce digital signals. These RSSI measurements can be used for localization.

5.1.2 Stargate

Stargate[26], shown in Figure 5.1, is a resource-rich node that provides more capabilities than the MICA motes. It is a powerful Linux based single board computer with Intel 400MHz X-Scale® processor (PXA255), Compact Flash, PCMCIA, Ethernet, USB Host, 64 MB SDRAM and an additional interface to communicate with a mote. We use a Stargate as the computational substrate for the mobile-robot. The Stargate runs the Robust Extended Kalman Filter which is implemented in Java. RSSI readings are measured for the mote interfaced with the Stargate, communicating with other sensors.

5.1.3 Lego MindStorm: Mobility

We use the popular Lego MindStorm[13] platform, shown in Figure 5.1 to emulate a mobile robot. It is a programmable, non-maneuvering robot that constructed by connecting small Legos together. A Stargate is mounted onto the Lego. An Infra Red tower is used to program the RCX box, which controls the Lego MindStorm. At
the core of the RCX is a Hitachi H8 microcontroller with 32K external RAM. The microcontroller is used to control three motors, three sensors, and an infrared serial communications port. Both the driver and firmware accept and execute commands from the PC through the IR communications port. To calculate the velocity, the Lego MindStorm is programmed to move at a constant speed (selected from eight power options) in a straight line.

(a) Mica2 mote (b) Stargate (c) Lego Mindstorm Robot

Figure 5.1: Hardware Components of the Hybrid Testbed.

Figure 5.2: Indoor sensor distribution and robot navigation topology.
5.2 Experimental Results

We placed the motes in the topology shown in Figure 5.2 and programmed the Lego Mindstorm robot to move in a straight line. We measured the distance travelled and time elapsed to accurately deduce the velocity of a mobile robot (sampling rate = 0.3s). The values of the weighting matrices are tuned with simulations using the modelled system parameters.

5.2.1 Computational Efficiency

To characterize computational efficiency, we measured the REKF computation time (for 10 sensors) as a function of the number of RSSI samples for the following cases:

- Java implementation (Sun’s JVM) on Pentium IV 3GHz machine
- Matlab implementation on Pentium IV 3GHz machine
- Java implementation, with Open-Wonka on Stargate 400MHz machine

The computation time in milliseconds is shown in Table 5.2. The performance of Matlab and Java running on Pentium IV 3 GHz does not differ much (in the same order). However for Stargate, the performance degrades significantly with the number of samples. The Stargate has very limited memory (64 MB SDRAM ) which makes Open-Wonka’s garbage collector inefficient. To improve the performance on
the Stargate, we can (i) break down the computation on Stargate to a smaller subsets of samples, or (ii) implement REKF in C instead of Java.

### 5.2.2 Estimation Accuracy

To evaluate estimation accuracy, we report on an experiment with four sensors, since the visualization of estimation convergence is clearer with a smaller number of sensors. The four sensors are positioned at (6.1, 6)m, (12.2, 13.6)m, (18.3, 21.2)m and (24.4, 13.6)m. The mobile vehicle (robot) is initially located at (0, 15.2)m and moves with a velocity of 2.52m/min. For our indoor implementation, with the collected data and modelling the RSSI, we use $p_{oi} = 160mw$ and $\varepsilon = 3$ (equation 3.2).

Figure 5.3 shows the localization of the four sensors with approximate error of 1m. The results are very close to the real positions. The Lego Mindstorm is a non-manoeuvring robot (constant velocity). By incorporating vehicle maneuver further improvements to the localization error can obviously be made.
Figure 5.3: (a) Sensor localization, and (b) localization error for the real system.

6 Algorithmic Improvements and Future Work

Experimental evaluation revealed the importance of finding the right parameters for the weighting matrices. Our next step is to implement automatic tuning of these parameters through machine learning techniques and enhance the localization scheme significantly. The algorithm stores all sensor states during the experiment period in order to validate the convergence of estimations. To improve computational efficiency, in a production system, where only the final estimation is needed, only two states for each sensor need to be stored (instead of nearly 500).

In general, the use of mobile robots in Delay-Tolerant Sensor Networks opens up a number of other interesting research possibilities. Once sensor localization has been performed, it is possible to create a topological map (or a path profile) that can
be optimized so that the data collecting mobile-robot can follow this path for data collection in the shortest possible time, or to meet storage and power requirements. Mobile robots could act as relays between disconnected portions of the network, thereby forming a relay network. The trajectory of a mobile robot can be dynamically recalculated so that a mobile robot can slow down whenever it needs to download a lot of information. We intend to explore these aspects in future work.

7 Conclusions

In this paper, we have provided a scheme for node localization using mobile robots in a delay-tolerant sensor network (DTN). To the best of our knowledge no other study has been done for such a network. DTNs are commonly deployed in long-term environmental monitoring applications. In a DTN, node localization need not happen in real time. Using one or more mobile robots to compute the location of sensors allows us to tradeoff computational time for node localization for several other benefits. First, we can eliminate processing constraints for small sensor devices. We can employ more sophisticated algorithms since processing is performed by the robot rather than sensor devices. Second, we can reduce the number of static location reference beacons required, by exploiting mobility of the robot. Third, it makes RSSI-based localization more viable. Because the robot-receiver is mobile, over a period
of time, we can statistically eliminate the fading noise in RSSI measurements.

We proposed applying a Robust Extended Kalman Filter based state estimator for node localization. It is computationally more efficient and robust to measurement noise compared to the more commonly used extended Kalman filter implementation. Real experiments in a large indoor area show that the localization accuracy is approximately 1m. This compares favorably with previously proposed RSSI localization schemes in an indoor setting[1] (accuracies within 3m), as well as with finer-grained acoustic time-of-flight localization schemes[19, 23] (accuracies vary 10cm - 25cm). Now that we have validated our ideas through simulation, implementation and experiment, we are working on further localization schemes refinements and on other mobile-robot uses in delay-tolerant sensor networks.

**Acknowledgements**

We would like to acknowledge Dang Xuan Thanh and Budi Mulyawan for the hardware implementation, experimental data collection and their insights to this research.
References


<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_{oi}$</td>
<td>600 mw</td>
<td>Base station transmission power</td>
</tr>
<tr>
<td>$N$</td>
<td>diag{10^{-2}, 10^{-2}, 10^{-2}, 10^{-2}, \ldots}</td>
<td>Weighting on the initial viscosity solution</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Q$</td>
<td>diag{10^{4}, 10^{4}, 10^{8}, 10^{8}, \ldots}</td>
<td>Weighting on the uncertainty in the user driving command</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$R$</td>
<td>diag{100, 3, 2 \times 10^{9}, 2 \times 10^{9}, 6 \times 10^{5}, 5 \times 10^{5} \times 65}</td>
<td>Weighting on the measurement noise</td>
</tr>
<tr>
<td>$T$</td>
<td>10 mins</td>
<td>Simulation time</td>
</tr>
<tr>
<td>$A_{max}$</td>
<td>50 m/s^2</td>
<td>weighting on ( u(t) )</td>
</tr>
<tr>
<td>$T_s$</td>
<td>0.65 s</td>
<td>Sampling interval</td>
</tr>
<tr>
<td>$x_1(0)$</td>
<td>[800m 2500m 0 0]'</td>
<td>1st sensor initial state</td>
</tr>
<tr>
<td>$x_2(0)$</td>
<td>[-5000m 0 0 0]'</td>
<td>2nd sensor initial state</td>
</tr>
<tr>
<td>$x_3(0)$</td>
<td>[-4500m 1500m 0 0]'</td>
<td>3rd sensor initial state</td>
</tr>
<tr>
<td>$x_4(0)$</td>
<td>[-3000m 2500m 0 0]'</td>
<td>4th sensor initial state</td>
</tr>
<tr>
<td>$x_5(0)$</td>
<td>[-1500m 4500m 0 0]'</td>
<td>5th sensor initial state</td>
</tr>
<tr>
<td>$x_6(0)$</td>
<td>[-2500m -500m 0 0]'</td>
<td>6th sensor initial state</td>
</tr>
<tr>
<td>$x_7(0)$</td>
<td>[-1500m -3000m 0 0]'</td>
<td>7th sensor initial state</td>
</tr>
<tr>
<td>$x_8(0)$</td>
<td>[-1000m -2250m 0 0]'</td>
<td>8th sensor initial state</td>
</tr>
<tr>
<td>$x_9(0)$</td>
<td>[-1700m 1000m 0 0]'</td>
<td>9th sensor initial state</td>
</tr>
<tr>
<td>$x_{10}(0)$</td>
<td>[3000m 900m 0 0]'</td>
<td>10th sensor initial state</td>
</tr>
<tr>
<td>$x_c(0)$</td>
<td>[2500m 2500m 2m/s 100m/s]'</td>
<td>mobile - robot initial state</td>
</tr>
</tbody>
</table>

Table 7.1: Simulation parameters for Scenario II.