A Wireless Sensor Network for Real-time Indoor Localisation and Motion Monitoring

Lasse Klingbeil
Autonomous Systems Laboratory
CSIRO ICT Centre
QCAT, Brisbane, Australia
Lasse.Klingbeil@csiro.au

Tim Wark
Autonomous Systems Laboratory
CSIRO ICT Centre
QCAT, Brisbane, Australia
Tim.Wark@csiro.au

Abstract

This paper describes the development and deployment of a wireless sensor network for monitoring human motion and position in an indoor environment. Mobile sensor nodes comprising mote-type devices, along with inertial sensors are worn by persons moving inside buildings. Motion data is preprocessed onboard mobile nodes and transferred to a static network of seed nodes using a delay tolerant protocol with minimal radio packet overhead. A Monte Carlo based localisation algorithm is implemented, which uses a person's pedometry data, indoor map information and seed node positions to provide accurate, real-time indoor location information. The performance of the network protocols and localisation algorithm are evaluated using simulated and real experimental data.

1 Introduction

Wireless sensor networks (WSN) are fundamentally a tool for measuring the spatial and temporal characteristics of almost any phenomena. An implicit aspect in obtaining spatial information is knowledge of the relative or absolute location of a point of measurement. Knowledge of node location is also crucial information for tasks such as tracking mobile objects through networks [12], development of spatial sampling algorithms or geographic routing protocols [15]. The ultimate goal of sensor networks is to remove the need for extensive network topology planning and thus the need for knowledge of node location when deploying. Whilst GPS [27] or other manual surveying techniques are available for determining node locations, the ability for nodes to self-localise greatly broadens the ease and scope of applications for which sensor networks can be used.

Another major driver for localisation within sensor networks is driven by networks with mobile nodes. The ability to track the position of nodes, e.g. vehicles, animals or people, opens up a wide range of applications in transport management, agriculture [31], military and health domains. Mobile nodes can also form an important role as data ferries in networks with disconnected nodes. As a result, localisation has been an important and growing research topic for wireless sensor networks [25, 13].

1.1 Related Work

Previous work in sensor network localisation has typically fallen into two categories: (i) fine-grained localisation and (ii) coarse-grained [17]. *Fine-grained algorithms*, are usually based on some detailed information such as an estimate of the distance or angle between two nodes. Given a minimum of three seed nodes, triangulation approaches can be used to estimate the location of a node relative to the location of anchor nodes.

In order to achieve fine-grained localisation, a sensor node typically requires specialised hardware and often extensive computational resources. Disadvantages include increased hardware costs, high energy usage and increased physical sizes of nodes. A summary of the most popular fine-grained approaches are given below:

- Radio Signal Strength (RSS): One of the simplest approaches that has been used for estimation of distances between nodes is receiver signal strength. An example of this type of approach is the RADAR system [1] where signal strength from static nodes is used to roughly track mobile indoor nodes.

- Time Difference on Arrival (TDoA): Range between nodes can be estimated far more accurately by estimating the time difference between transmission and reception of slow travelling signals such as ultrasound or acoustic waves. By transmitting both radio and ultrasound/acoustic waves at the same time, a receiver

978-0-7695-3157-1/08 $25.00 © 2008 IEEE
DOI 10.1109/IPSN.2008.15

node has a transmission reference time stamp by which to calculate time difference. The obvious downside of this approach is the need for nodes to have additional hardware for dealing with acoustic signals. An example of the use of this approach is in the use of "Recent technology" where ultrasonic signals are used in a peer-to-peer fashion to determine the relative location of mobile computing devices [4].

- Angle of Arrival (AoA): Whereas the previous two approaches have been based on distance estimates between nodes, location can also be derived from knowledge of the angle of arrival between nodes. Techniques for obtaining angle estimates include the use of phased arrays of RF or ultrasonic receivers or by using rotating directional beacons [21].

In contrast to the previous methods, coarse-grained localisation approaches focus on using a minimal amount of information to derive a location estimate [11]. Common approaches are given below:

- Proximity: The simplest of all location techniques is that of proximity measurements — that is, a simple decision as to whether two nodes are within radio range of each other. The decision can be made based on whether any packets are received at all, or can be based on a threshold approach requiring some consistent connectivity over time. A good example of this is in the use of RFID tags for node localisation [32].

- Centroid: In the case of proximity measurements from multiple reference nodes, then location can be estimated by simply calculating the centroid of the known reference node locations [5].

- Approximate point in triangle (APIT): In this technique [11], a set of regions are defined by forming sets of triangles from all seed nodes which are in the proximity of an unknown node location. The unknown location can then be determined as the intersection of the triangular regions.

Approaches using light have also been investigated as means of localising a network of nodes in a centralised fashion. In the StarDust system [28], nodes containing small reflectors allow their location to be determined when light is shed over the entire network area. Other more sophisticated centralised schemes include the use of self-organising maps (SOM) to concurrently estimate node locations given hop counts over entire network [10].

An additional field that has driven progress in the area of mobile localisation is the field of robotics [9, 7, 19]. A robot can be thought of as a mobile sensor node with extended computational ability and controllable state. Mobility is a subset of sensor network research that holds particular importance for localisation [13] with applications ranging from target tracking [12] through to personal dead-reckoning systems for GPS-denied environments [22, 24, 2]. As a result there are many learnings that can be taken from the robotics community and applied to the field of sensor networks localisation.

1.2 Motivation

The vast majority of previous work in localisation within sensor networks has focussed on theoretical and simulation approaches to validation of algorithms [25, 20, 18, 10]. There has only been limited work undertaken in actual implementation of real-time, range-free localisation methods within sensor networks. Whilst there is a large body of experimentally validated techniques developed from within the robotics community for mobile robot localisation, these have only seen limited investigation within sensor networks applications to date.

This paper describes the implementation and evaluation of a protocol for real-time, mobile-node localisation where a Monte-Carlo based approach is used to combine a local mobility model and indoor map information, with proximity information from static seed nodes. The system has been designed as a cheap, effective solution for personal-tracking in indoor environments. The key advantages of the system we propose is that excellent localisation accuracy can be achieved with low cost inertial sensors and radio transceivers, minimal system calibration and low network traffic.

Our key contributions in this paper are as follows:

- Novel combination of local mobility model, range-free localisation and map information

- Evaluation of an actual, real-time implementation of a WSN localisation network

2. System Platform

2.1. Hardware Platform

The hardware platform we have used for this work is part of a family of devices known as 'Fleck'. Inspired by the original Berkeley mote, since 2002 we have developed three generations of the devices known as the Fleck-1, Fleck-2 [26] and most recently, the Fleck-3 [14]. These devices incorporate a number of key design features which make the platform ideal for a wide range of applications including outdoor, long-term deployments, which has been a key focus of much of our work to date [30].

The Fleck-3 uses the Atmega128 micro-controller along with the Nordic NRF905 radio transceiver operating in the

915MHz band. The platform also incorporates power management circuitry to allow solar charging of batteries in the case of outdoor use. The device also incorporates a real-time clock chip to reduce micro-controller overheads. The architecture relies heavily on the SPI bus where the Atmega128 acts as the SPI master and can communicate with the radio, the flash memory, the real-time clock over the SPI interface. The real-time clock and the radio can both interrupt the Atmega128 to signal alarms, packet transmission and packet reception.

An additional feature of the Fleck-platform is the ability to easily expand by adding additional sensors or sensor interface boards, as well as coprocessor boards such as a DSP. For this particular application we have developed a custom inertial movement unit (IMU) daughterboard which stacks directly on top of the Fleck-3 as shown in Figure 1.

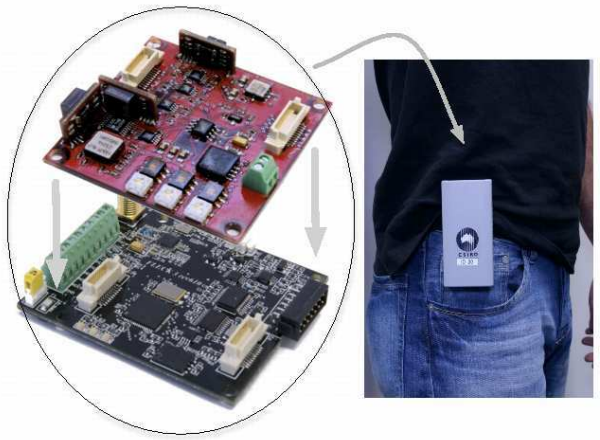

Figure 1. Fleck3 sensor board (bottom) with optional IMU daughterboard (top), containing accelerometers, magnetometers and gyroscopes.

The IMU daughterboard contains three single-axis gyroscopes (ADXRS150), two dual-axis accelerometers (ADXL202E) and a tri-axial magnetometer (HMC1053). The sensors are sampled using a 16 bit ADC where the sampling rate is limited by the data processing on the microcontroller.

2.2. Software Platform

We have also developed our own multi-threaded operating system known as "Fleck OS" (FOS) [6] which is closest in spirit to MANTIS OS [3]. As others have reported [16, 8], the event-driven programming paradigm does not scale well with program size, leading to difficulty in developing and maintaining large applications.

FOS provides a priority-based, non-preemptive (cooperative) threading environment with separate stacks for each thread, which has the advantage of providing a simple concurrent programming model which does not require semaphores. The scheduler is also responsible for CPU power management and enters the lowest mode consistent with thread resource requirements. Time-critical operations such as analog data sampling or high-speed timers are handled by interrupt-level callbacks.

3. Network Topology and Protocol

This section describes the network backbone for the localisation system. The network consists of two key parts, referred to as the *static* and the *mobile* networks. The static network is comprised of the "seed" nodes — i.e. the nodes which act as anchor points for the entire system. The mobile network comprises the nodes which are to be localized. They are designed to be worn by persons moving through a building and are comprised of the Fleck-3 platform and IMU daughterboard as described in Section 2.1 and shown in Figure 1. The overall system architecture of these static and mobile nodes is illustrated in Figure 2.

3.1 Static Network Design

The static network comprises a number of seed nodes which are installed at fixed and known locations in a region of interest, for example one node per room, or multiple nodes in bigger rooms. A single base node is connected to a gateway computer for final processing and visualization. A key role of the static network is to transfer all the data packets coming from the mobile network back to the base node. In the network used for the experiments in this paper, we used a single-hop protocol with an acknowledgment Medium Access Control (MAC) layer. This ensured fast and reliable transfer with low network overhead, but also obviously limited the size of localisation area, in our case to around a 50m radius from the base node.

The other key role of the static network is to provide a sufficient number of anchor points for the localisation algorithm to perform robustly. As described later in Section 5, being close to a seed node is used for a coarse-grain position measurement of the mobile node. To increase the value of this information, the transmission range of a mobile node has to be as small as possible, but still large enough to have a seed node in range most of the time while walking through the network. Empirical test showed, that using the radio chip without antenna and using the -2dB transmission power level reduce the range to a value in the order of 5m (Figure 3). These data are taken under 'ideal' conditions, where there is free line of sight between both nodes. While walking through a building the values might change a lot

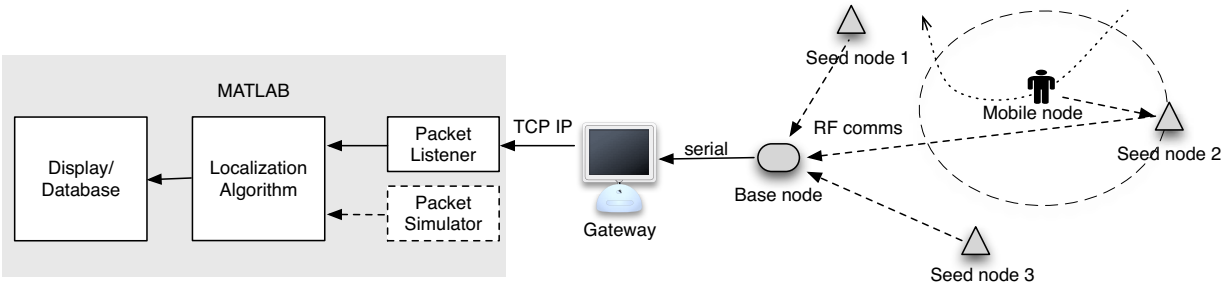

Figure 2. System architecture for indoor localisation system.

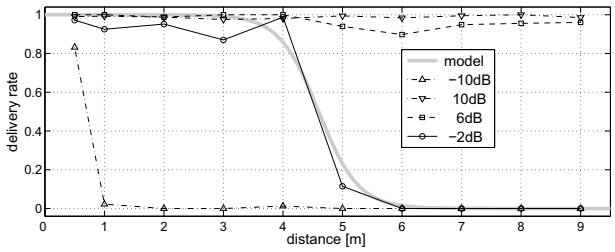

Figure 3. Illustration of observed radio throughput between a static wall nodes and mobile node.

due to occlusion by walls and the body of the person. We use a modified sigmoid function to model the transmission probability depending on the distance:

$$f(d) = 1 - \frac{1}{1 + e^{-\sigma_r(d-\hat{r_t})}} \quad (1)$$

where $\hat{r_t}$ is the nominal range and σ_r is an arbitrary parameter for range noise. This model is later used to represent uncertainty of the proximity based position information in the localisation algorithm.

3.2 Mobile Network Design

The mobile network, comprising nodes worn by people, has two important roles. Firstly, the number of data packets being transmitted from the mobile nodes needs to be minimized to avoid congestion in the static network and save energy. This is achieved by preprocessing the motion data onboard the device to reduce to a minimal number of relevant events. This process is described in detail in Section 4. Secondly, these events have to be reliably transferred to the static network despite the reduced connectivity, mentioned above. This is achieved by the use of a *delay tolerant network* (DTN) [33] protocol where data is buffered locally until a seed node is available for upload. It should be mentioned here that an overflow of the buffer is not an issue in our system. The buffer has the ability to store 100 walking step events. Assuming a step frequency of about 1-2Hz it allows the person to walk around for about a minute without connection to the static network, before the buffer is full. In that case old events are overwritten.

The delivery of packets from mobile to static nodes is assured using an *implicit acknowledge* mechanism. If a static node receives a packet from a mobile node and forwards it back to base, the forwarded packet is also received by the mobile node and triggers the upload of the next packet. In this way we guarantee a maximum transmission rate between the mobile and the static network with minimal packet overhead.

Figure 4 shows the main threads running on a mobile node. Every event generated from the mobile sensor data is locally stored in an event queue. At the same time, the mobile node beacons at a predefined interval, using the first event in the queue as a beacon message. If a nearby static node receives this message, it forwards it back to base. The forwarded message is also received by the mobile source node, which takes this as notification of packet delivery and keeps on going with the next event in the queue if there is any.

The local buffering of event data introduces an inherent latency between the occurrence of a relevant motion event and the availability of that event information to the localisation algorithm running at the network base. The amount of latency depends on the density of the seed nodes, the link quality between mobile and static nodes and the beaconing interval for broadcasting data messages. This is discussed in more detail in Section 6.2.

The localisation algorithm makes use of proximity information, *only* for events which occur within the proximity of a static node. As such, the position measurement given by the forwarding seed node can not automatically be assigned to all motion events in the data message — only those which actually occur within the vicinity of the seed node. To han-

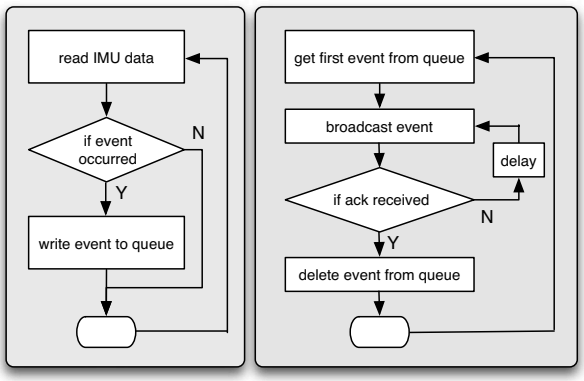

Figure 4. Illustration of threads running on mobile nodes.

dle this problem, we maintain a *nearest node* variable in the mobile node, which is set by an implicit acknowledge message and reset to a *NULL* value after one second without acknowledgements. This variable is also stored in the queue as part of the motion event. It should be noted here, that a mobile node data packet can be forwarded (and therefore acknowledged) by multiple seed nodes. Every received acknowledgement updates the nearest node's information on the mobile node, but the fact of receiving the same message via different seed nodes is not used for localisation.

4. On Board Data Processing

A crucial aspect of the system is preprocessing of motion data on mobile nodes. Preprocessing the raw data and extracting only relevant events reduces the number of packages that need to be sent over the network. Given the computational resources on a mobile node are clearly limited, the processing algorithms we use need to be highly resource efficient.

The relevant information we extract for the proposed localisation method is (i) the occurrence of a walking step together, (ii) movement direction. The algorithms used to derive this information is presented below.

4.1. Step Detection

To detect the occurrence of a walking steps, we use the data from the accelerometer sensor where the sampling frequency is set to 25Hz. The acceleration signal vector magnitude (svm) is corrected by the offset due to gravity and then filtered using a simple FIR averaging filter with a history of 8 samples. Step events are then detected using a custom algorithm (outside the scope of this paper) based on thresholds and some heuristics to ensure robustness. Figure 5 illustrates the functionality of the step detection. The

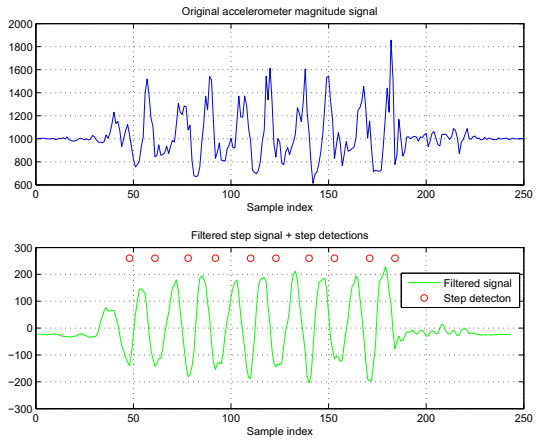

Figure 5. Illustration of the step detection derived from accelerometer.

upper plot shows the unfiltered acceleration svm, while the lower plot shows the filtered signal and the detected steps. The parameters of the algorithm are tuned to maximize the detection accuracy of a certain person in a normal walk mode. Analysis and optimization for different people, walking styles and speeds are beyond the scope of this paper.

4.2. Heading

For every detected step, the corresponding heading information is also calculated. The heading can either be calculated by integrating the z-axis angular rate sensor data (parallel to the gravitational axis) or by using the x and y-axis components of the Earth's magnetic field.

Both methods, used alone, have drawbacks. Magnetic field sensors are very sensitive to disturbance from nearby metallic objects. In indoor environments especially, this can lead to large errors in heading calculations. Conversely, when integrating gyroscope data, along with needing a known start value, measurement errors can accumulate very quickly. Assuming that gyroscopes have a reasonable short term stability however, and that magnetometers are only disturbed over short periods of time, we can combine the advantage of both types of sensors by using a *complementary filter* with a fixed weight W as:

$$h_k = (1 - W)(h_{k-1} + \omega_k dt) + W h_{mag,k} \qquad (2)$$

where h_k is the heading estimation, ω_k the gyroscope reading and $h_{mag,k}$ the heading value based on only the magnetometer reading.

In our experiments we found that a weighting factor of 0.01 gives sufficient accuracy. So the filter relies mostly on

the gyroscope data, but the magnetometers have still enough influence to compensate for gyroscope drift and unknown starting angles. A detailed analysis of the accuracy of the heading estimation is beyond the scope of this paper, however in Figure 6 we show its functionality in a critical situation. In this example, a person wearing a mobile node

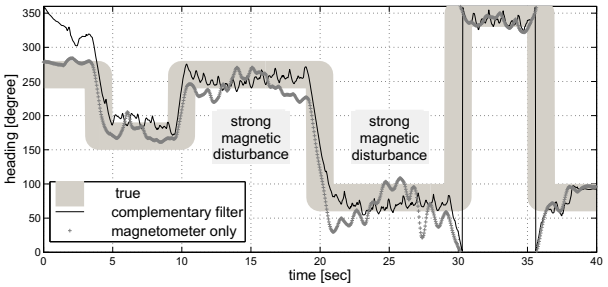

Figure 6. Illustration of the heading estimation based on magnetometer and gyroscopes.

walks around two 90 degree corners, passes a large magnetic field disturbance, then turns around and walks back the same way. The actual (ground-truth) heading is indicated by the wide grey line. From the Figure 6 we can observe two things. Firstly, the estimator starting with a wrong heading angle converges correctly towards the true heading. This takes about 10 seconds in the example case. Secondly, the estimator is not affected by the strong magnetic field, disturbing the magnetometer readings. We show in later simulations, that the accuracy of the localisation algorithm is not critically dependent on the accuracy of the heading data.

5. Localization Algorithm

This section describes the algorithm which is used to perform the localisation of mobile nodes. Our system is based around the use of Monte Carlo filtering, which has been used in some previous localisation work [13, 25]. Our algorithm extends previous work however, by combining three key pieces of information in the filter being:

- Proximity information from static seed nodes
- Improved mobility information derived from onboard inertial sensors
- Indoor map information

5.1. Recursive Bayesian Estimation

Bayesian estimation methods are widely used to estimate a systems state based on noisy sensor information. We summarize the concepts here, for detailed overview with a focus on localisation, see [29].

In Bayesian estimation methods, the state $\mathbf{x}_k$ of system at a time t_k is estimated by calculating the *a posteriori* probability $p(\mathbf{x}_k|\mathbf{z}_0,\ldots,\mathbf{z}_k)$, where $\mathbf{z}_0,\ldots,\mathbf{z}_k$ are all measurements about the system up to t_k. The *a posteriori* probability can be calculated using Bayes' Rule:

$$p(\mathbf{x}_k|\mathbf{z}_0,\ldots,\mathbf{z}_k) = \frac{p(\mathbf{z}_0,\ldots,\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k)}{p(\mathbf{z}_0,\ldots,\mathbf{z}_k)}, \quad (3)$$

which can be transformed into a recursive equation:

$$\begin{aligned} p(\mathbf{x}_k|\mathbf{z}_0,\ldots,\mathbf{z}_k) &= \frac{p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}_0,\ldots,\mathbf{z}_{k-1})}{p(\mathbf{z}_k|\mathbf{z}_0,\ldots,\mathbf{z}_{k-1})} \\ &= \eta \cdot p(\mathbf{z}_k|\mathbf{x}_k)p(\tilde{\mathbf{x}}_k). \end{aligned} \quad (4)$$

where $\eta = 1/p(\mathbf{z}_k|\mathbf{z}_0,\ldots,\mathbf{z}_{k-1})$ is a normalization factor. The term $p(\mathbf{z}_k|\mathbf{x}_k)$ is called the *measurement model* and relates the state of the system to its observations, including knowledge about uncertainties of the sensors. In our implementation the system variable will be the position of a mobile node and the observation will be the position of a nearby seed node.

The term $p(\tilde{\mathbf{x}}_k) = p(\mathbf{x}_k|\mathbf{z}_0,\ldots,\mathbf{z}_{k-1})$ is called the *a priori* probability and is calculated as:

$$p(\tilde{\mathbf{x}}_k) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{z}_0,\ldots,\mathbf{z}_{k-1})\mathrm{d}\mathbf{x}_{k-1} \quad (5)$$

The term $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ is called the *process model* and it describes the knowledge (including uncertainties) about the evolvement of the system over time. In our implementation this model will incorporate odometry data from the mobile node as well as information about the indoor environment.

Starting from an initial probability $p(\mathbf{x}_0)$, each recursive update of a Bayesian filter, as described above, can be performed in two steps:

Prediction The *a priori* probability is calculated from the last posterior probability using the process model (Equation 5).

Correction The new *a posteriori* probability is calculated using the prior and the measurement model (Equation 3).

If the probability distributions are Gaussian and the models are linear, a *Kalman Filter* provides an optimal and efficient implementation of Bayesian filters. If the models are non-linear, they are often linearized using the *Extended Kalman Filter*. In the *Unscented Kalman Filter* the probability distribution is represented by a carefully chosen set of points in the state space, which conserves the Gaussian properties of the distribution under a non-linear transformation.

5.2. Sequential Monte Carlo Filters

In Monte Carlo sampling based techniques, a probability distribution $p(\mathbf{x})$ is represented by a number of N weighted samples $\mathbf{x}^{[i]}, i = 1..N$, with weights $w^{[i]}$ as:

$$p(\mathbf{x}) \approx \sum_i w^{[i]} \delta(\mathbf{x}^{[i]} - \mathbf{x}). \quad (6)$$

Sequential Monte Carlo filters are also called *Particle Filters*, where samples are referred to as *particles*. Starting from the initial probability $p(\mathbf{x}_0)$, which may be represented as equally distributed samples with equal weights, the recursive update is performed as follows:

In the *prediction step* every sample $(\mathbf{x}_{k-1}^{[i]}, w_{k-1}^{[i]})$ of the *a posteriori* distribution $p(\mathbf{x}_{k-1}|\mathbf{z}_0, \ldots, \mathbf{z}_{k-1})$ is replaced by a new sample according to the process model $p(\mathbf{x}_k|\mathbf{x}_{k-1})$, which leads to a new set of samples $(\tilde{\mathbf{x}}_k^{[i]}, \tilde{w}^{[i]})$ representing the *a priori* distribution.

In the *correction step*, the weight $w^{[i]}$ of every sample of the *a priori* distribution, is updated according to the measurement model:

$$w^{[i]} = \tilde{w}^{[i]} \cdot p(\mathbf{z}_k|\tilde{x}_k^{[i]}), \quad \sum_i w^{[i]} = 1. \quad (7)$$

Together with the normalized weights, the prior set of samples now approximates the *a posteriori* probability. In the last step (resampling), a new set of samples is drawn with replacement from the prior set with the probability of a sample being drawn given by its weight factor. The final set represents the new posterior as well, but now the samples are equally weighted. At this point we can start with a new prediction step.

During the resampling, unlikely samples are omitted which leads to many duplicates in the final set. To avoid ending up with only a single sample, noise is introduced during the prediction step to separate samples that have the same values.

5.3. Monte Carlo Localization

In this section we describe the implementation of the particle filter we use to estimate the position of a mobile node within a static network of seed nodes. The role of the process model is to describe the transition from a state $\mathbf{x}_{k-1}$ at the time t_{k-1} to state $\mathbf{x}_k$ at the time t_k. In our case the system is a person wearing a mobile node and its state is given by its position $\mathbf{x}_k = (p_x p_y)^T(t_k)$. Thus the process model is given by a motion model, which describes our knowledge about the movement of the person.

A common model, which also has been used in [13, 25], is to assume that a person is moving in random directions with random speed. The speed is equally distributed between 0 and a maximum velocity v_{max}. Other models assume a constant velocity and model the uncertainty by inducing a small random acceleration in every prediction step. In that case the state variable also contains the speed of the person.

A key advantage of the system described in this paper is that we determine information about the motion of a person from local inertial sensors. Every step that is detected, combined with the heading measurement, provides information about a relative position change. This data can then be incorporated into the prediction model as *control inputs* $\mathbf{u}_k$.

The other source of information we utilize during the prediction step is the indoor map of the building. A map provides a natural constraint on the paths a person can take and allows the motion model to check if a certain step is possible — i.e. a person can't walk through walls! Our motion model in the prediction step (Eq. 5) is then written as $p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{u}_k, m)$, where m is the map. In our case, the control consists of the heading angle ϕ_k and the stride length l_k. To simplify the motion model, the stride length is currently not measured and therefore assumed to be constant.

In the actual implementation of the localisation algorithm, discussed in Section 6.2, every sample representing a possible position of the mobile node is recalculated as soon as a new radio packet, indicating a detected walking step, is received at the base node as:

$$\begin{pmatrix} p_x \\ p_y \end{pmatrix}_k^{[i]} = \begin{pmatrix} p_x \\ p_y \end{pmatrix}_{k-1}^{[i]} + \hat{l}_k^{[i]} \cdot \begin{pmatrix} \sin(\hat{\phi}_k^{[i]}) \\ \cos(\hat{\phi}_k^{[i]}) \end{pmatrix} \quad (8)$$

As mentioned earlier, it is important to introduce noise into the model to avoid degeneracy of the filter. This is done in a natural way due to the uncertainties n_ϕ, n_l of the heading and stride length information as:

$$\hat{l}_k^{[i]} = l_k + n_l^{[i]} \quad , \quad n_l^{[i]} \text{ drawn from } p(n_l) \quad (9)$$

$$\hat{\phi}_k^{[i]} = \phi_k + n_\phi^{[i]} \quad , \quad n_\phi^{[i]} \text{ drawn from } p(n_\phi) \quad (10)$$

The knowledge about the map is incorporated by setting the weight of a samples to zero, if it crossed a wall during the prediction step (Equation 8).

$$\tilde{w}_k^{[i]} = \begin{cases} w_{k-1}^{[i]} & \text{no wall crossed in (8)} \\ 0 & \text{wall crossed in (8)} \end{cases} \quad (11)$$

In the correction step we utilize any available information about the proximity of the mobile node to a static seed node. This can be seen as an absolute position measurement (position of the forwarding node) with a large error (transmission range of the mobile node). The measurement model in (Equation 7) can be written as:

$$p(\mathbf{z}_k|\tilde{x}_k^{[i]}) = p(X_n|\tilde{x}_k^{[i]}) \propto f(d_n^{[i]}), \quad d_n^{[i]} = X_n - \tilde{x}_k^{[i]} \quad (12)$$

where X_N is the position of the forwarding static node and $f(d)$ is a function describing the transmission probability between the mobile and the static node depending on their distance. In the simplest case this could be defined as:

$$f(d) = \begin{cases} 1 & d \leq \hat{r_t} \\ 0 & d > \hat{r_t} \end{cases} \quad (13)$$

assuming a transmission range $\hat{r_t}$. Due to the transmission irregularities discussed earlier, we chose the more appropriate model given in Equation 1. The correction step is only executed, if the step was detected within the range of a static node, otherwise the filter continues with the next prediction step leading to further spreading of the samples.

After updating and renormalising the weights (Equation 7), the resampling step follows, where a new set of samples is drawn with replacement using the weights as a drawing probability. The weights are set to an equal value and the position of a person is estimated using the mean of all the samples. The spread of the samples (e.g. the standard deviation in both dimensions) is then a measure for the confidence of the estimation.

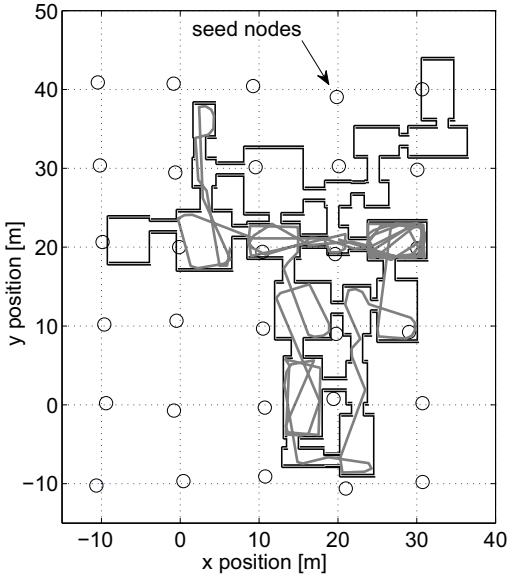

Figure 7. Example path generated by the simulator

6. Evaluation

6.1. Simulation Results

To fully evaluate the performance of the localisation algorithm prior to final implementation, we implemented a custom simulation environment. The simulator was designed to allow randomly generated paths to be created within a randomly generated building map [23], similar to the environment we undertook our experiments in. A simulated person could then move along the generated path where inbuilt radio models would allow packets to be generated and sent between mobile and static nodes arriving back at the base node.

To generate a path, we assume a person starts from a given starting position and then walks in a certain direction by adding up steps, whose lengths are drawn from a random distribution. When the person hits a wall, the simulator assigns a random turn angle and then continues on. This algorithm has been optimized in a way that every trajectory represents a realistic path a person would walk through the building. (Figure 7).

As a metric for the simulation we use the estimation error, which is the distance between a reconstructed position on the graph and the true one. The estimation error of a whole path is the root mean square (rms) of the errors along the path. We ran every simulation setting with a number of different random generated paths and show the standard deviation of the estimation errors as error bars in the plots. For some simulations we also show the maximum estimation error.

The following assumptions and parameters are used for the simulations:

On board data processing: The step detection probability is 100%. The heading measurements have a Gaussian noise with $\sigma_h = 10°$. There is an additional bias added to the heading measurement, which depends on the current position and varies between $-10°$ and $10°$ over a distance of $10m$. Thus an event marked as *close to a seed node* depends on the seed density and the radio transmission model.

Network properties: The seed nodes are distributed on a grid, where the grid distance can be varied as a simulation parameter (see Figure 7 with seed distance of 10m). If a message from a mobile node is received by a seed node is determined by the *Degree of Irregularity*[1] parameter δ introduced in [34]. The transmission range r_t for every message is drawn from a uniform distribution between $(\hat{r_t} - \delta\hat{r_t})$ and $(\hat{r_t} + \delta\hat{r_t})$. A message gets transmitted if the distance between two nodes is smaller than r_t. The nominal range $\hat{r_t}$ is set to 5m for all simulations, and the DoI parameter δ is set to 0.3 unless otherwise noted.
Additionally a certain percentage of all events is dropped to simulate packet loss in the static network.

Localization algorithm performance: We use the same parameters for the localisation algorithm for all simulations. Unless otherwise noted, we use a number of 500 particles and use the true starting point of the path as a starting point

[1] The DoI is originally meant to describe radio irregularities in the transmission direction, but we apply the model to irregularities over time.

for the reconstructed path.

Using the simulator, we can demonstrate how the use of the motion model and map information increases the accuracy of the location estimation. Since the benefit is expected to be larger for lower seed densities, we plot the rms error and the maximum estimation error against the seed distance. This comparison is shown in Figure 8. The random motion

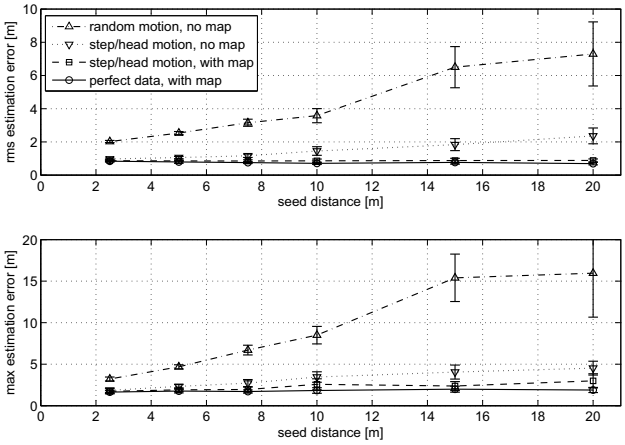

Figure 8. Estimation error for different models with varying seed distance.

model is comparable to the model used in [25] and already described in Section 5.3. The moving direction is assumed to be unknown and the moving distance per time step is restricted by a maximum velocity. As expected, the rms error increases with increasing seed distance. We also notice a saturation of the error towards higher distances. This is due to the fact, that the paths generated by the simulator are in a constrained area, which limits also the maximum estimation error.

The heading based motion model is shown with and without incorporating the map information. The error still increases, however much less than with the random direction model. By also incorporating the map information, the error is nearly constant up to a seed distance of 20m. The solid line shows the error with perfect data, that means the heading measurement is assumed to be known exactly.

The variation of the seed distance also changes the seed density, which is defined as the average number of seed nodes in range of a mobile node. This has a direct influence on the amount of traffic in the network, since a message sent by a mobile node is transferred back to base by every seed node in range. Figure 9 shows the seed density for varying seed distance. From about 10m the data packets are forwarded by only one node which minimizes the network traffic, but still gives reasonable localisation accuracy (Figure 8).

We also undertook simulation experiments varying the

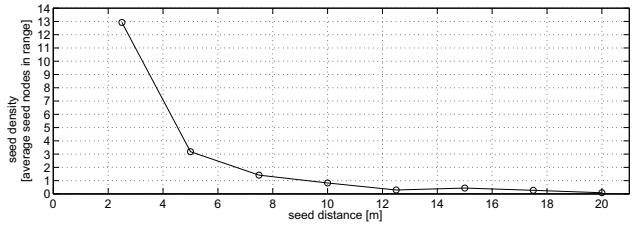

Figure 9. Seed density depending on the seed distance.

number of particles used by the localisation algorithm. These results are shown in Figure 10.

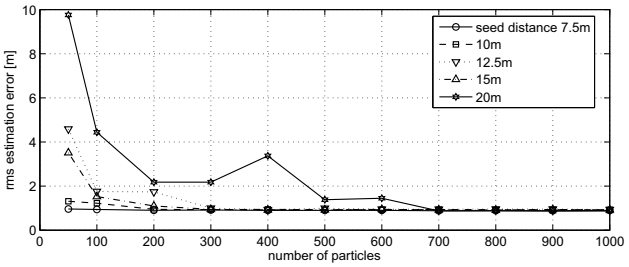

Figure 10. Estimation error for different seed distances, varying the number of particles in the localisation algorithm.

It is observed that the estimation accuracy does not improve for particle numbers bigger than 400 if the node distance is up to 15m. Smaller particle numbers may give reasonable results in most cases, but have an influence on the robustness of the filter. This can be seen clearly in the next simulation shown in Figure 11.

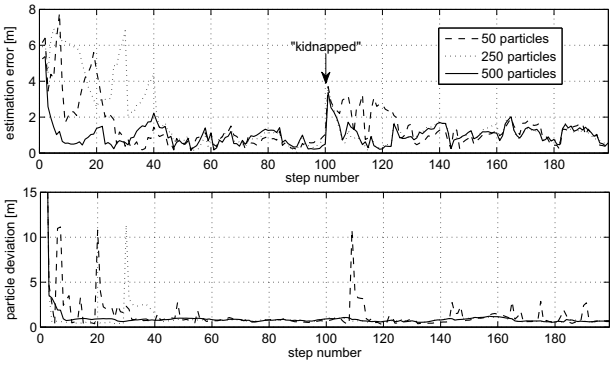

Figure 11. Estimation error over time, using different numbers of particles.

In Figure 11, the initial position of a person (step 0) is unknown and particles are therefore spread over the whole map. The filter, using 500 particles, converges to the true position after about 10 steps, while the cases with 250 and 50 particles take about 40 steps. At step 100, the true path is deliberately cut (the person is *kidnapped*) and started again in a different room. It can be seen that the algorithm needs about 10 steps to find the new position. The lower graph of Figure 11 shows the spreading of the particles, which is calculated by adding the squared standard deviations of the particles x and y components.

Figure 12 shows how the localisation algorithm is affected by varying network properties. In this case the seed density is set to 10m and the number of particles to 500. We

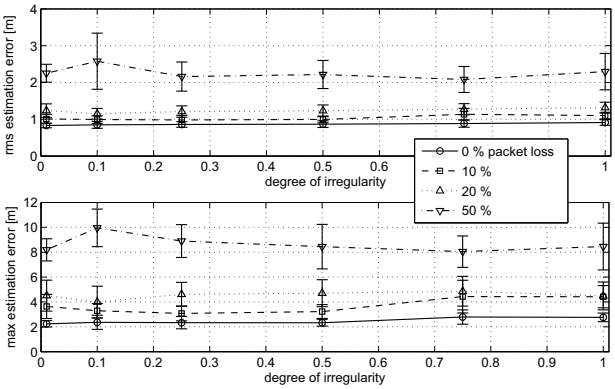

Figure 12. Estimation error for different packet losses, varying the transmission irregularity.

observe that the estimation error is nearly independent of the degree of irregularity of the radio transmission between the mobile and the seed node. This demonstrates a key advantage of the proposed algorithm, since it is not dependent on correct radio propagation models like RSS based algorithms.

6.2. Experimental Results

In this section, we show experimental results taken from a sensor network deployed around our office building. Nine seed nodes were deployed on various positions throughout an area in the building as shown in Figure 13. The distance between the nodes is about 5m to 10m. A person wearing a mobile node on the waist walked along a path indicated by the thick grey line. The messages generated by the mobile node are forwarded back to base using the protocol described in Section 3 and processed by the localisation algorithm running on the base computer. The black crosses show the reconstructed steps using the motion model, the

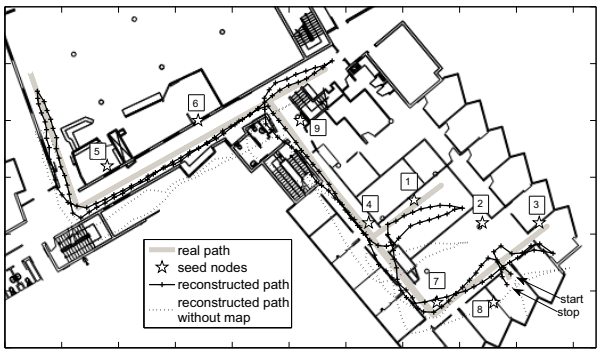

Figure 13. True and reconstructed walking path using the heading based motion model

seed node positions and the building map. The dotted line shows the reconstructed path without using indoor map information.

Since we do not have a reference system, we manually created the true path as a polygon and defined the error metric as the closest distance to the closest polygon segment. This metric results in a slight underestimation of the true error, since it only gives the deviation *orthogonal* to the path and not *along* the path. However, as can be seen in Figure 13, this is a reasonable estimate if the reconstructed path is not varying much from the true path. Plot 14(d)

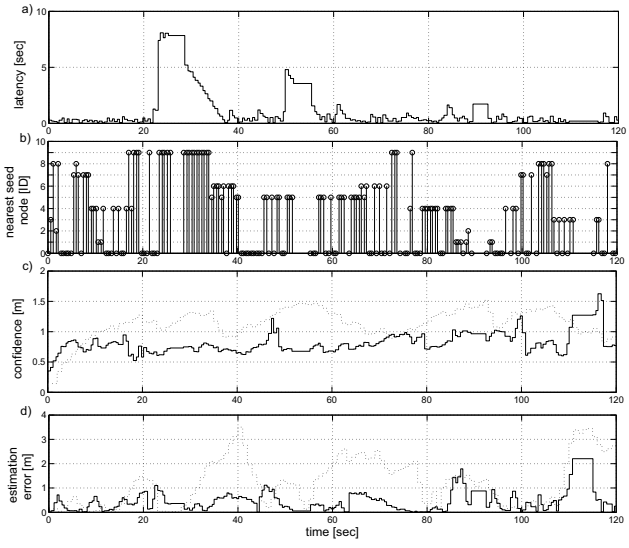

Figure 14. Latency, near node ID, estimation error and estimation confidence over time.

shows the errors over time for estimation with and without map information. Plot 14(c) shows the confidence of the po-

sition estimation, which is given by the standard deviation of the particles in the x and y-direction.

Latency is another important metric to evaluate in sensor networks. For this application we define latency as the time between the occurrence of a walking step and the a successful transfer from the mobile node to the static network. The transfer time from a seed node to the base node is assumed to be virtually zero since the static nodes form a fully connected network and were running at a 100% duty cycle for these experiments. The latency thus shows how long an event is stored in the buffer of the mobile node and depends both on the time of disconnectedness of the mobile node and on the retry time after an unsuccessful upload. Plot 14(a) shows that the latency is about a second most of the time, with a maximum value of 8 seconds during a longer time with bad connectivity. Plot 14(b) shows which mobile node has been used in the measurement model of the localisation algorithm. A node ID of zero means, that the step was detected out of range of any seed nodes.

The indoor walking experiment was repeated over a longer time period of about half an hour, walking the same path multiple times. Figure 15 shows the probability density distribution for both the latency and the reconstruction error calculated from the experiments. We define the overall error

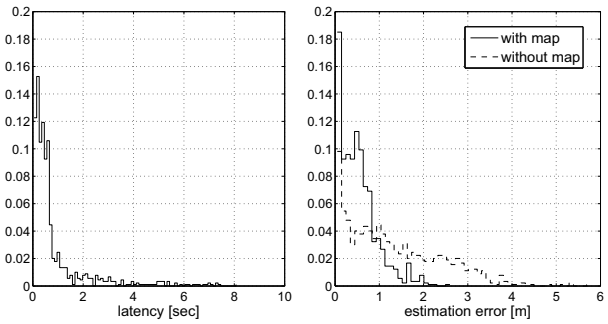

Figure 15. Empirical probability density distribution for latency and estimation error calculated from multiple experiments.

and latency as the standard deviation from zero. The overall latency is 1.3 seconds with a maximum value of 8 seconds. Assuming an uncertainty of 1m on the true path, the overall estimation error with an algorithm using the map information is 1.2m with a maximum of about 2.5m, and 2m with a maximum of 5.2m without the map. This is consistent with the simulation results in Figure 8.

7. Conclusions and Future Work

7.1 Conclusions

We have presented an indoor localisation system for a wireless sensor network, which estimates the position of mobile nodes worn by persons moving through a network of static seed nodes with known positions. Relevant information about the motion of the person is generated by the mobile nodes and transferred to a central PC, where they are combined with knowledge about the indoor environment and the seed node positions using Monte Carlo based estimation algorithms. Data packets are processed as soon as they arrive at the central PC, which enables real-time tracking of a person while moving through a building.

We have evaluated the performance of our localisation system through both simulations and experimental validation. Our simulations show that the algorithm is robust against radio transmission irregularities and heading estimation errors which occur from sensor noise and magnetic disturbances. In a deployment in our office building we showed, that a person walking through the area could be localized with a rms error of about 1m and a maximum error of about 3.5m. The network latency was about 1 second rms and 8 seconds maximum.

7.2 Future Work

There are several ways to improve the system into the future. One is to extend the tracking area by using multihop routing techniques in the static network. Another interesting avenue is to generate more information from the IMU data than just step events and heading values. We are working on additional research in accelerometer-based activity classification to detect events such as sitting, lying or falling, which allows the system to be used for health care applications like monitoring of elderly people in care facilities. Future work will also investigate ways to estimate the step *length* of a walking person to improve localisation accuracy as well as detect new types of events such as stair climbing which then enables position estimation over multiple floors.

Another direction for future work is decentralization of the system. In the current implementation the localisation algorithm is running on a central computer, where all the data from the network are is collected via the base node. In principle however, every mobile node has all the information available that is needed to estimate its own position. Instead of forwarding a mobile node event message to the base node, a seed node would simply send an acknowledge message containing also its own position information. Given this scenario, other nodes of the mobile network could also be used as seed nodes.

References

[1] P. Bahl and V. N. Padmanabhan. Radar: An in-building rf-based user location and tracking system. In *INFOCOM*, 2000.

[2] S. Beauregard and H. Haas. Pedestrian dead reckoning: A basis for personal positioning. In *Proceedings of the 3rd Workshop on Positioning, Navigation and Communication*, 2006.

[3] S. Bhatti, J. Carlson, H. Dai, J. Deng, J. Rose, A. Sheth, B. Shucker, C. Gruenwald, A. Torgerson, and R. Han. MANTIS OS: An Embedded Multithreaded Operating System for Wireless Micro Sensor Platforms. *Mobile Networks and Applications*, 10(4):563–579, 2005.

[4] U. Bischoff, M. Strohbach, M. Hazas, and G. Kortuem. Constraint-based distance estimation in ad-hoc wireless sensor networks. In *EWSN*, 2006.

[5] N. Bulusu, J. Heidemann, and D. Estrin. GPS-less low cost outdoor localisation for very small devices. *IEEE Personal Commucations Magazine*, pages 28–34, 2000.

[6] P. Corke. Fos - a new operating system for sensor networks. In *Proceedings of the 5th European Conference on Wireless Sensor Networks (EWSN08)*, 2008.

[7] P. Corke, R. Peterson, and D. Rus. Localization and navigation assisted by cooperating networked sensors and robots. *The International Journal of Robotics Research*, 24(9):771–786, Oct. 2005.

[8] C. Duffy, U. Roedig, J. Herbert, and C. J. Sreenan. An Experimental Comparison of Event Driven and Multi-Threaded Sensor Node Operating Systems. In *Proceedings of the Third IEEE International Workshop on Sensor Networks and Systems for Pervasive Computing (PERSENS2007), White Plains, USA*. IEEE Computer Society Press, Mar. 2007.

[9] D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte carlo localization: Efficient position estimation for mobile robots. In *Sixtenth National Conference on Artificial Intelligence (AAAI-99)*, 1999.

[10] G. Giorgetti, S. K. Gupta, and G. Manes. Wireless localization using self-organising maps. In *IPSN*, 2007.

[11] T. He, C. Huang, B. M. Bium, J. A. Stankovic, and T. Abdelzaher. Range-free localization schemes for large scale sensor networks. In *ACM MobiCom'03*, 2003.

[12] T. He, S. Krishnamurthy, J. A. Stankovic, T. Abdelzaher, L. Luo, R. Stoleru, T. Yan, and L. Gu. Energy-efficient surveillance system using wireless sensor networks. In *MobiSys*, 2004.

[13] L. Hu and D. Evans. Localisation for mobile sensor networks. In *Tenth International Conference on Mobile Computing and Networking (MobiCom)*, 2004.

[14] J. Karlsson, T. Wark, P. Valencia, M. Ung, and P. Corke. Demonstration of image compression in a low-bandwidth wireless camera network. In *IPSN*, 2007.

[15] B. Karp and H. Kung. Gpsr: Greedy perimeter stateless routing for wireless networks. In *MobiCom*, 2000.

[16] O. Kasten and K. Römer. Beyond event handlers: programming wireless sensors with attributed state machines. *Proceedings of the 4th international symposium on Information processing in sensor networks*, 2005.

[17] B. Krishnamachari. *Networking Wireless Sensors*. Cambridge University Press, 2005.

[18] L. Lazos and R. Poovendran. Serloc: Robust localisation for wireless sensor networks. *ACM Transactions on Sensor Networks*, 1(1):73–100, 2005.

[19] L. Liao, D. Fox, J. Hightower, H. Kautz, and D. Scultz. Voronoi tracking: Location estimation using sparse and noisy sensor data. In *Proc of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2003.

[20] X. Nguyen, M. I. Jordan, and B. Sinopoli. A kernel-based learning approach to ad-hoc sensor network localisation. *ACM Transactions on Sensor Networks*, 1(1):134–152, 2005.

[21] D. Niculescu and B. Nath. Ad hoc positioning system (aps) using aoa. In *INFOCOM*, 2003.

[22] L. Ojeda and J. Borenstein. Personal dead-reckoning system for gps-denied envionments. *International Workshop on Safety, Security and Rescue Robotics*, 2007.

[23] S. O'Sullivan. Map viewer - a mapping tool for mobile robotics. In *http://www.skynet.ie/ sos/mapviewer/main.php*, 2007.

[24] A. Raj, A. Subramanya, D. Fox, and J. Bilmes. Rao-blackwellized particle filters for recognizing activities and spatial context from wearable sensors. In *The 10th International Symposium (ISER 2006)*, 2006.

[25] M. Rudafshani and S. Datta. Localization in wireless sensor networks. In *Information Processing in Sensor Networks (IPSN)*, pages 51–60, 2007.

[26] P. Sikka, P. Corke, P. Valencia, C. Crossman, D. Swain, and G. Bishop-Hurley. Wireless adhoc sensor and actuator networks on the farm. In *IPSN*, pages 492–499, 2006.

[27] R. Stoleru, T. He, and J. A. Stankovic. Walking gps: A practical solution for localisation in manually deployed wireless sensor networks. In *EmNets*, 2004.

[28] R. Stoleru, P. Vicaire, T. He, and J. A. Stankovic. Stardust: A flexible architecture for passive localization in wireless sensor networks. In *SenSys'06*, 2006.

[29] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT-Press, 2005.

[30] T. Wark, P. Corke, P. Sikka, L. Klingbeil, Y. Guo, C. Crossman, P. Valencia, D. Swain, and G. Bishop-Hurley. Transforming agriculture through pervasive wireless sensor networks. *IEEE Pervasive Computing*, 6(2):50–57, 2007.

[31] T. Wark, C. Crossman, W. Hu, Y. Guo, P. Valencia, P. Sikka, P. Corke, C. Lee, J. Henshall, J. O'Grady, M. Reed, and A. Fisher. The design and evaluation of a mobile sensor/actuator network for autonomous animal control. In *IPSN*, pages 206–215, 2007.

[32] P. Wilson, D. Prashanth, and H. Aghajan. Utlizing RFID signaling scheme for localization of stationary objects and speed estimation of mobile objects. In *IEEE International Conference on RFID*, 2007.

[33] Z. Zhang. Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: Overview and challenges. *IEEE Communications Surveys and Tutorials*, 8(1):24–37, 2006.

[34] G. Zhou, T. He, S. Krishnamurthy, and J. A. Stankovic. Models and solutions for radio irregularity in wireless sensor networks, 2006.