Wireless sensor networks for human intruder detection

The SmartDetect Project Team[a]

Indian Institute of Science, Bangalore, India

Abstract | In this paper we report on the outcomes of a research and demonstration project on human intrusion detection in a large secure space using an ad hoc wireless sensor network. This project has been a unique experience in collaborative research, involving ten investigators (with expertise in areas such as sensors, circuits, computer systems, communication and networking, signal processing and security) to execute a large funded project that spanned three to four years. In this paper we report on the specific engineering solution that was developed: the various architectural choices and the associated specific designs. In addition to developing a demonstrable system, the various problems that arose have given rise to a large amount of basic research in areas such as geographical packet routing, distributed statistical detection, sensors and associated circuits, a low power adaptive micro-radio, and power optimising embedded systems software. We provide an overview of the research results obtained.

[a]SmartDetect is a research and demonstration project funded by the ER&IPR Division of the Defence Research and Development Organisation (DRDO), Government of India, during the period 2006-2010. The following project team contributed to the work presented in this paper: Anurag Kumar (Principal Investigator), P. Vijay Kumar (co-Principal Investigator); (the following are in alphabetical order within category) **Faculty Investigators:** Bharadwaj Amrutur, G.K. Ananthasuresh, Navakanta Bhat, R.C. Hansdah, Malati Hegde, Joy Kuri, Vinod Sharma, Y.N. Srikant, Rajesh Sundaresan; **Project Staff:** Tarun Agarwal, S.V.R. Anand, Pallav Bose, Vijay Dewangan, Shalini Keshavamurthy, A.V. Krishna, Pavan Kumar, Sharath Kumar, D. Manjunath, Sundeep Patil, Poornima V.L., K. Aditya Prasad, Santosh Ramachandran, Anurag Ranjan, Subathra Sampath, Jeena Sebastian, Vishwas Vasuki; **Students:** Pranav Agrawal, Sahebrao Sidram Baiger, Vinod Kumar Chouhan, Taposh Banerjee, Satyam Dwivedi, Abhishek Gupta, Santosh Hedge, Neeraj Kumar, Prachee Jindal, Premkumar Karumbu, Sambuddha Khan, Girish Krishnan, Syam Krishnan, Chaitanya U. Kshirasagar, K.P. Naveen, Mohan Rathod, Deepak Ravi, U. Raviteja, R. Abu Sajana, Ramanathan Subramanian, Thejas, Amulya Ratna Swain, Lalitha Vadlamani, Leena Zacharias.

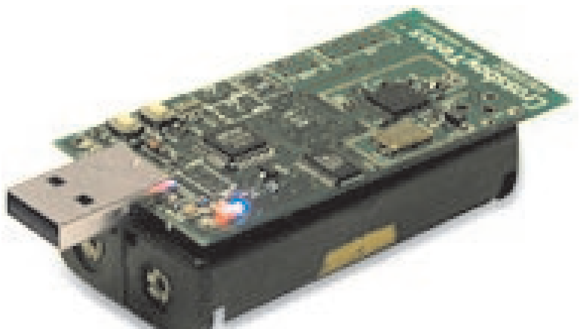
1. Introduction

This paper provides an overview of a multidisciplinary, multifaculty project carried out in the Indian Institute of Science in the area of wireless sensor networks for the detection of human intruders into secure regions, such as the grounds of high security buildings, a large industrial installation, or an airport.

A smart wireless sensor device comprises miniature sensors, a low power microprocessor, a simple digital radio transceiver, and a small battery, all in a compact package. It is expected that such devices (commonly referred to as *motes* after the prototypical device developed at the University of California, Berkeley) will become so power efficient, that in conjunction with energy harvesting technologies, and energy efficient algorithms, their batteries could last for years. Figure 1 shows the TelosB mote (developed and marketed by Crossbow Technology [75]) which was the device on which

Mote: A smart wireless device, comprising miniature sensors, a low power microprocessor, a simple digital radio trasceiver, and a small battery, all in a compact package.

Figure 1: The Crossbow TelosB "mote" which was utilised for developing the SmartDetect WSN.



SmartDetect: The name of the wireless sensor network system that we developed for intrusion detection inside a secured area.

the SmartDetect system described in this paper was developed.

In a typical application, such devices would be deployed in the 100s or 1000s, in a planned or random fashion (being strewn onto a large tract of land from an airplane, or embedded into building structures as they are constructed). On being initialised, the devices would discover each other, and then self-organise into a multihop wireless (packet) network. They would then begin to sense the environment, and use various distributed algorithms to schedule packet transmissions between themselves, and carry out various communication and local computation tasks, e.g., to detect and identify an event (such as a fire or an intruder) and communicate this to an operator at the edge of the network. Thus, these systems can be viewed as easily deployable, self-configuring, *embedded distributed smart instrumentation*.

These systems can be seen to depend on the following key technical elements:

1. Efficient and low cost microcontrollers, sensing and energy harvesting devices;
2. Support software (i.e., operating systems and compilers) that is simple (yet provides the required primitives), compact, energy efficient and power aware;
3. Distributed and energy efficient algorithms for self-organisation, scheduling of packet transmissions, locationing, time-synchronisation, sensor data processing (such as quantisation, data compression, detection, estimation, identification, classification, and tracking), and system security.

The specific application addressed in this project was one of securing from human intrusion the periphery and grounds of a large building, industrial installation, or an airport. The perimeter of the geographical area that needs to be secured could be several kilometers (e.g., a 1 km square area). The activities in such a situation would normally be limited to a few centrally located buildings, car parks, driveways. However, large parts of the grounds would see little activity, and may only sporadically be patrolled. The problem, therefore, is to deploy a wireless sensor network in order to detect quickly and to locate any abnormal activity in the normally inactive grounds areas. Such networks would clearly need to be long-lived. Another problem is that of securing the grounds around a building for a short time-period when some sensitive activity is taking place in the building (e.g., the visit of an international dignitary to a hotel or guest house). Such a sensor network could be deployed by a special task force entrusted with the security of the visitor, and the network would be removed and stored once the visitor leaves.

Among the major projects that also addressed similar objectives, two notable ones are "A Line in the Sand" project (see [6] and [7]) and the VigilNet Project (see [31]). "A Line in the Sand" was the name given to a field experiment conducted by the NEST team of Ohio State University under the DARPA-NEST program. The experiment involved the deployment of a 90 node wireless sensor network with 78 magnetic sensors, and 12 additional radar sensors. A major contribution of the work was that it demonstrated the feasibility of discrimination between object classes using a network of binary sensors. The VigilNet project, executed at the University of Virginia, used magnetic sensors to detect and track the position of moving vehicles. Node power management was projected as the major strength of this effort. This was performed by what the authors call "sentry service component" that selected a subset of motes called sentries to monitor events. The remaining motes were allowed to remain in a low-power state until an event occurred. Both these projects used Mica2 motes [75] under the TinyOS operating system. For the operation of the various protocols used in these projects, a tight time synchronization and maintenance of neighbour information were essential. Both these requirements are hard to meet, given the harsh outdoor environment where node failures are common, and the large network diameter. Also, message security, which is extremely important for this kind of application, was not considered. Our major contribution is a design for SmartDetect that takes into account the above issues and makes the solution robust, reliable, scalable and secure. SmartDetect does not assume tight time synchronization, does not maintain any neighbour information, and has security features built into it.

Sensing modality: The type of signal from the environment that is sensed by the device. Examples include acoustic, magnetic, optical, thermal, etc.

The SmartDetect project has been a unique experience in collaborative research. It has brought together the expertise of ten faculty members of varied interests (sensors and micromechanical systems, circuits, computer systems, communication, networking, signal processing, and security), in order to execute a large research and demonstration project that spanned three to four years. After the identification of the team and selection of the application area, the project evolved as follows: (i) Key technical problems were identified; (ii) Pieces of the puzzle, related to each faculty's expertise, were taken, analysed, and solutions were developed; (iii) Various solutions were implemented, tested, and rejected or revised. The project groups met twice a month over a period of three to four years; once each month to discuss among themselves, and once to present relatively complete work to the entire project team, in which meeting the DRDO collaborators from CAIR (the Centre for Artificial Intelligence and Robotics) also participated. A website that contained the minutes of all project meetings and a repository of referred to and generated literature, was maintained. Broadly speaking, the project has resulted in two main outcomes:

- A demonstrable wireless sensor network for human intrusion detection, built on the commercially available TelosB motes (see Figure 1), with the human sensing modality being *passive infrared (PIR)*.
- A large amount of basic research leading to publications in journals and conferences, and training (in full or in part) of 10 PhD students, 6 MSc (Engg.) students, 8 ME students, and over two dozen project staff.

In this paper we provide an overview of the techniques that went into the development of the SmartDetect system, and also an overview of the related basic research that was conducted in areas such as sensors, low power radios for motes, geographical routing, distributed detection, and power optimising systems software.

The following is the section-wise outline of the paper. In Section 2, we first provide a comparative overview of sensor technologies. For this project, we found passive infra-red (PIR) sensors to be the most effective. The remainder of Section 2 describes the sensor platform design, and the signal processing techniques used to infer human presence from the PIR sensor signals. In Section 3 we discuss issues, and our solutions, related to the wireless mesh network that connects the PIR sensor platforms to the base station. Topics discussed are network self-organisation and geographical forwarding. The processing environment on the motes is severely limited, with a very simple operating system and limited memory. In Section 4 we describe the architecture of SmartDetect software, and some of the implementation challenges we faced. Being cheap devices, motes also have inaccurate clocks that drift relative to each other. In order to have even a crude common notion of time, time synchronisation becomes necessary; our approach to addressing this issue is provided in Section 5. In applications such as intrusion detection, the wireless sensor network is faced with adversaries who jeopardise correct behaviour. Security protocols are therefore necessary to thwart such attempts. Section 6 discusses SmartDetect's robustness to certain security attacks.

Our developmental efforts towards the SmartDetect platform have not only resulted in algorithms and design insights, but have also led to a considerable amount of new research results. In Section 7 we discuss the research on sequential event detection in sensor networks. Multimodal sensor fusion is a possibility that one could explore in the future. To this end, and with footsteps detection in mind, a MEMS accelerometer, and its associated capacitance measurement electronics, have been developed; this is reported in Section 8. Innovative techniques are needed to reduce the energy consumption of mote components. In Section 9 we report the development of a novel adaptive radio, that automatically adjusts its power consumption depending on the quality of the received signal. Operating systems provide applications with a convenient interface to the hardware and compilers convert user programs to machine code in an efficient manner. Research on an energy efficient operating system, and energy optimising compilers is presented in Section 10.

2. Detecting human intrusion

2.1. *Intrusion sensors: An overview*

Human intrusion can be detected using many sensor modalities [6]. Some of the relevant ones are listed and compared in Table 1 (adapted from [6]). Six types of sensors are included in the table. All of these are passive in the sense that, unlike radar or ultrasonic sensors, they do not emit a signal and sense how targets modify it. Passive sensors are preferred in sensor networks where there is limited energy. Magnetic sensors assume that the intruder, such as an armed person, has magnetically sensitive material. Ferromagnetic material creates a specific magnetic signature that can be detected using a magnetometer. Any metallic content worn by the intruder can be detected using electromagnetic techniques. Seismic and acoustic sensors are based on the vibrations caused by the intruder. Both come under the general category of vibration sensors which we describe next in some detail as a preamble to our prototyping effort highlighted in Section 8.

Seismometers and geophones measure ground displacements. Velometers are devices that measure velocities.

Vibration-based surveillance sensors can be classified into two major groups, namely, acoustic sensors and motion sensors. Acoustic sensors measure the sound produced by the entity that is to be detected or monitored. In the case of vehicles, the main sources of sound are engine and power-train noise, track/tyre noise and exhaust noise. Footsteps of humans and animals, fluttering of wings by birds, etc., also generate sound in addition to the entity's vocal sound. Sensors that measure sound are essentially microphones and hydrophones. On the other hand, vibratory motion sensors sense displacement, velocity and acceleration using seismometers/geophones, velometers and accelerometers, respectively. The physical construction of both classes of sensors is almost the same: they contain a spring-restrained mass which inevitably will have some damping. However, the frequency, range of operation and resolution of these sensors will be significantly different. Their cost also varies depending on their level of sophistication. It is unlikely that one sensor would work for detecting/monitoring varied sound/vibration sources.

Additionally, in the case of heavy vehicles there might be coupling between the acoustic noise and ground vibrations. The acoustic waves travel at different speeds and their amplitudes decrease at different rates with distance or get absorbed at different rates. This helps in distinguishing the type of vehicle or other noise source. Thus, in a surveillance application both acoustic and vibration sensors are needed. A good example of this can be found in an extensive study called "Bochum Verification project for Military Vehicle Detection" [3] that was conducted to identify vehicles in different environments. The vehicles included in this survey were cars, small and large trucks, armored personnel carriers and battle tanks. This study used two different types of microphones, one type of accelerometer and one type of geophone in multiple numbers. While an experiment on a tarmac road on sandy soil needed 1 accelerometer, 6 geophones and 2 microphones, an experiment on a concrete road on weathered layer of old lava needed 27 geophones and 4 microphones. With multiple sensors, and signal processing, the direction of arrival can be identified.

Optical and thermal sensors work on the principle of disturbance in the line of sight of the sensors. Humans, animals, and vehicles have 'hot spots' or specific thermal signatures that distinguish them from vegetation and buildings, and enable detection. Chemical sensors rely on particular chemical species associated with the intruder. Humans do leave a chemical trail but detecting it requires the sophistication of trained dogs and warrants an array of specialized sensors that can detect many chemical species [6].

The criteria for comparison shown in Table 1 were chosen keeping in mind their use in a sensor network. The comparison is subjective and depends on specific characteristics of particular sensors.

It is important that a sensor has sufficiently long range so that the density of the sensor motes can be kept reasonably low. Magnetic, thermal, and chemical sensors however have limited range and hence are less favoured than the other types.

When a sensor is used in a network for intrusion detection, it is not enough to give a signal in the event of intrusion; it is necessary to process that information in order to avoid false alarms. For example, in a vibration sensor, not only the magnitude of the ground vibration but also the spectral (i.e., frequency related) information is necessary to discern a disturbance as an intrusion. Sophisticated processing of the signals is needed. A good sensor is one that requires the least processing.

Packaging and mounting or deploying an array of sensors is also an important consideration. Seismic sensors require more care than other types because the stiffness of their mounting significantly affects their performance. Acoustic sensors, being sensitive to external vibrations, also are not favourable in this regard.

Sensitivity to line-of-sight obstructions can be either good or bad depending on the application. But usually, a sensor that can detect in spite of an obstruction in between the sensor and the intruder is favoured. A good example is a magnetic sensor that is not affected by vegetation in between. Vibration sensors too are not affected by an obstruction in the line-of-sight. On the other hand, optical, thermal, and chemical sensors get affected to various degrees by stationary or moving obstructions.

It is useful to have a sensor indicate the direction in which the intrusion has occurred. Magnetic and thermal sensors cannot usually sense this directionality of the intrusion.

In view of the limited energy available to a sensor mote in a network, it is beneficial to have a provision for energy harvesting. Only vibration sensors are amenable to this. The same sensor element might be used to harvest vibration energy in the case of seismic and acoustic sensors.

The last criterion listed in Table 1 is very important. It is preferred that a sensor performs

Table 1: Qualitative comparison of sensors for human intrusion detection. Here '+' denotes desirable and '–' denotes undesirable. (Adapted from [6])

Sensor type	Is it suitable for medium to long range detection?	Extent of signal processing required:	Level of complexity of packaging and mounting:	Is it sensitive to the line of sight?	Is it sensitive to the direction of the intrusion event?	Is energy harvesting possible?	Is it affected by the ambient conditions?
Magnetic	No (-)	High (-)	Low (+)	No (+)	No (-)	No (-)	No (+)
Seismic	Yes (+)	Medium (±)	High (-)	No (+)	Yes (+)	Yes (+)	Somewhat (±)
Acoustic	Yes (+)	Medium (±)	High (-)	No (+)	Yes (+)	Yes (+)	Somewhat (±)
Optical	Yes (+)	High (-)	Low (+)	Yes (-)	Yes (+)	No (-)	Yes (-)
Thermal	No (-)	High (–)	Low (+)	Yes (-)	No (-)	No (-)	Yes (-)
Chemical	No (-)	High (-)	Low (+)	No (+)	No (-)	No (-)	Somewhat (±)

consistently in all ambient conditions such as low or high winds, in bright light or in darkness, in sunlight or in rain, etc. Only magnetic sensors are good in this regard as compared with others.

Based on the foregoing discussion, it is apparent that a single sensor might not completely meet the needs of a surveillance application. It should also be noted that commercial sensors, even if they are readily available and are within budget, may still pose packaging problems and may need considerable customization.

Passive InfraRed (PIR) sensor: A sensor that measures the infrared light emanating from objects. Motion detectors usually use PIR sensors.

It is pertinent to note that military and environmental surveillance based on vibration sensing using an array of micromachined accelerometers or microphones has attracted the attention of several groups. For example, Draper Labs has developed an array of biologically inspired array of microphones for localizing the direction of sound within two degrees [20,55]. Applied MEMS Inc. [5], has an Unattended Ground Sensor (UGS) module that uses micromachined accelerometer. Their device demonstrated that human footsteps could be picked up for monitoring purposes. Hougen et al. [36] used a vibration sensor as part of a miniature robotic system. Furstenau et al. [24] developed a vibration/acoustic sensor system for monitoring passage of vehicles, their direction and type in an airport environment.

In this project, we chose passive infrared (PIR) sensors for initial demonstration of the efficacy of the algorithms and their implementation. In view of long-term viability, seismic sensors were chosen for indigenous development. This was because they score well on most criteria set forth in Table 1. However, the sensitivity and bandwidth requirements are quite stringent when they are used in a surveillance application to detect human footsteps. Such sensors, even though available commercially, are either very expensive or do not meet the performance requirements. Hence, the development of high-resolution and high-bandwidth accelerometer work was undertaken as part of this project. This is described in Section 8.

2.2. *Utilizing off-the-shelf PIR sensors*

Taking performance, pricing and availability into account, the particular PIR sensor chosen for our project was the analog Panasonic motion sensor AMN24111 [62]. This sensor has four sensing elements arranged so as to form a 2×2 array (see Fig. 2) that is enclosed by a multilens made up of multiple contiguous plano-convex lenses [41,45].

Each single pixel is capable of capturing temporal variations in the temperature. The pixels are wired in differential mode to avoid triggering the sensor due to changes in the ambient temperature. Thus, any common temperature change simultaneously perceived by two pixels with opposite polarities fails to set off the sensor. Infrared (IR) radiation incident on each plano-convex lens is focused onto the sensing elements of the PIR sensor. Thus, an array of virtual beams is created by the

Figure 2: The sensor and the quad pixels.

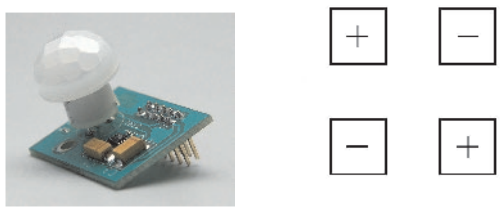


Figure 3: Cross-sectional and top view of the beams.

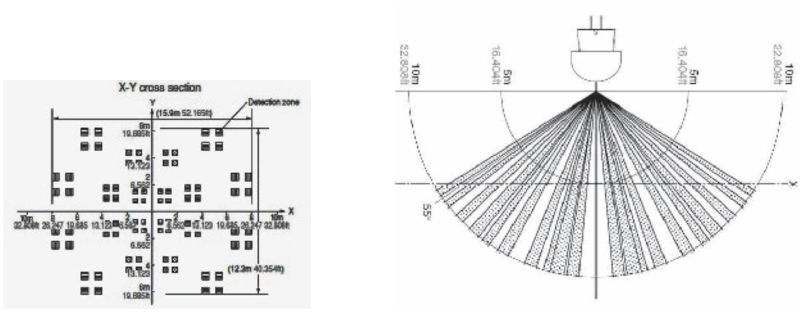
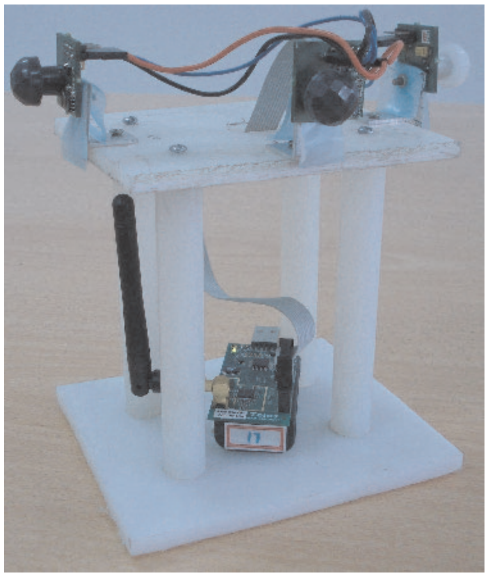


Figure 4: Three Sensor Platform.



multilens which forms the field of view of the sensor. We refer to these beams as the Virtual Pixel Array (VPA). The top-view and cross-section of the VPA at a distance of 5 metres are shown in Fig. 3.

Clutter: Unwanted signals, in the sensing modality, emitted by objects not of interest to the detection problem.

The sensor produces an electrical signal proportional to the difference in the rate of variation of incident–radiation–intensity between the two diagonals (see Figure 2). When an intruder moves at a uniform speed along a circular path around the sensor, the beams are cut at a uniform rate. Thus, the sensor's output resembles a triangular wave (sinusoidal after filtering) in which the number of half-cycles is same as the number of beams in the horizontal plane and the frequency observed is proportional to the angular velocity of the intruder[1]. The angular field view of each sensor is approximately 110° and thus in our experimental set-up, 3 sensors were mounted oriented at 120° relative to each other on a single platform, so as to obtain an omni-directional sensing range (see Fig. 2). The data from the three sensors were fed to the 3 ADC channels of the TelosB mote.

Support Vector Machine (SVM): A supervised learning technique that attempts to separate data points of one label from data points of another label, usually using a hyperplane in the data space.

The challenge and prior work

While the response at the output of a PIR sensor to a moving intruder is somewhat predictable, the principal challenge in using PIR sensors in an outdoor setting, is one of detecting intrusions reliably in the presence of wind-blown clutter while maintaining a low false-alarm rate. The manufacturers of commercially available sensors for instance, recommend careful placement of their PIR detectors to prevent false alarms resulting from vegetation, air currents, etc. In [80,27,8], the PIR signal is first high-pass filtered to remove low frequency components resulting from slow environment changes and the signal energy is then compared against an adaptive threshold. In particular, an unsupervised adaptation technique is used to adjust the energy threshold for target detection. In most of the other work in the open literature, decisions are made based on either simple thresholding of either the PIR signal itself or else, thresholding on the energy computed in a window.

Haar Transform (HT): A transform representation of the original data, analogous to the Fourier Transform. The HT has good time-frequency resolution.

[1]These sensors were intended by Panasonic to be used as motion detectors in indoor environments for applications such as automated lighting etc.

Our approach

We have developed a low-complexity Support Vector Machine (SVM) training-based algorithm that uses the Haar Transform (HT) to separate intruder from clutter. We believe that training of the form inherent in SVM should form an essential part of any solution, given the large variations in intruder and clutter signatures. Also, SVM makes possible a more fine-grain frequency-domain approach to separating intruder from clutter. While results are presented here only for the analog Panasonic Motion Sensor AMN24111, they are readily extended to other PIR sensors. Training and testing were carried out on actual experimental data and the algorithm was found to exhibit good performance. The low-complexity nature of the algorithm serves to achieve a key objective in a wireless sensor network, namely the extension of network lifetime.

2.2.1. *The detection algorithm*

We assume the intruder to be a human traveling in the vicinity of the sensor and use the term *clutter* to describe the sensor's output as a result of wind-blown vegetation.

Figure 5: Spectral signatures of intruder and clutter at the output of the digital and analog PIR sensors respectively.

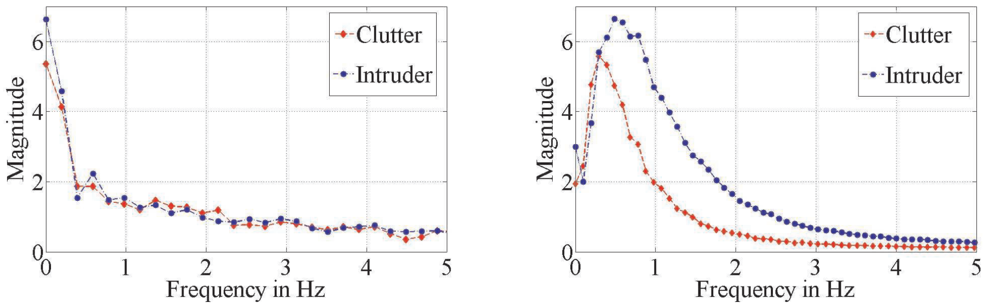
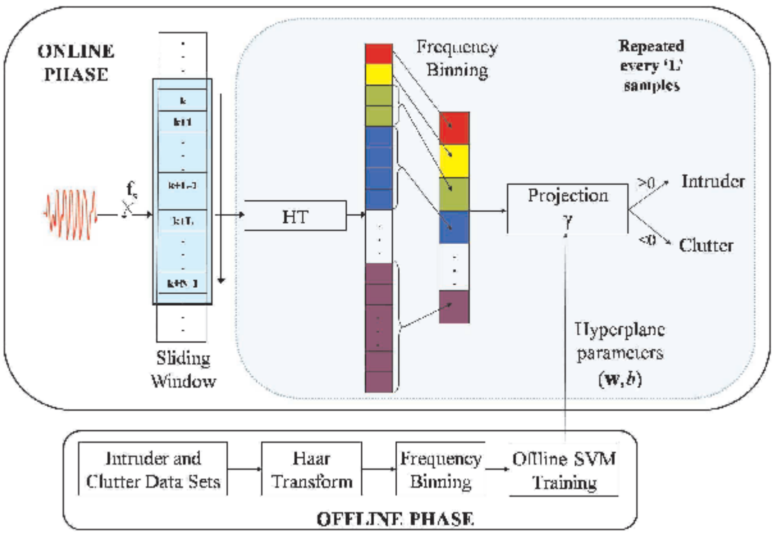


Figure 6: Functional block diagram of the algorithm.



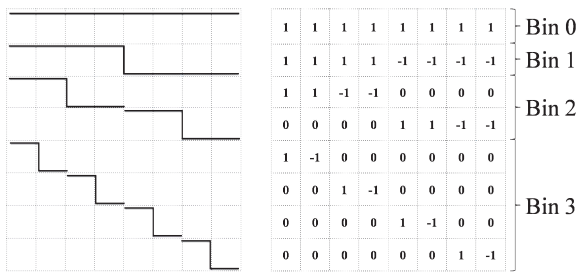
Overview of the algorithm

For the purpose of maximizing battery life, we decided to use the HT for computing the spectrum of intruder and clutter signals in preference to the computationally intensive Discrete Fourier Transform as only additions and subtractions suffice to compute the HT. An alternative would have been to use Walsh Hadamard Transform (WHT) but the HT was preferred as it can be computed with lower complexity, even when compared with the WHT. In addition, it has the ability to reuse past computed HT coefficients for the next window and can potentially be used to yield time-frequency localization information. A sampling frequency (f_s) of 12.5 Hz was chosen based on the frequency content of intruder and clutter waveforms (see Figure 5). A functional block diagram of the algorithm appears in Figure 6. A block of 128 (N) consecutive samples is transformed by the HT. The energy in each of these transformed components are binned into 8 frequency bins. The resultant binned vector is passed on to a classifier (obtained by off-line SVM training) which classifies it as either intruder or clutter. This entire process is repeated on a sliding-window basis, every 16 (L) samples.

The Haar transform and frequency binning

Since the Haar transform is wavelet based, coefficients are designed to provide both frequency and time localization information. As a result, the breakdown of 128 Haar coefficients is as follows: there is one coefficient assigned to frequency 0 (the DC component) and 2^k coefficients attached to signals of frequency $2^k, 0 \le k \le 6$. Thus, there are a total of $\log(N) + 1 = 8$ frequencies in all and in our algorithm, we collect together the energy in each of

Figure 7: Frequency bins corresponding to the 8-point Haar matrix.



$$\begin{array}{cccccccc|l}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & \text{Bin 0} \\
1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & \text{Bin 1} \\
1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 & \\
0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 & \text{Bin 2} \\
1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & \\
0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & \\
0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & \text{Bin 3} \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 &
\end{array}$$

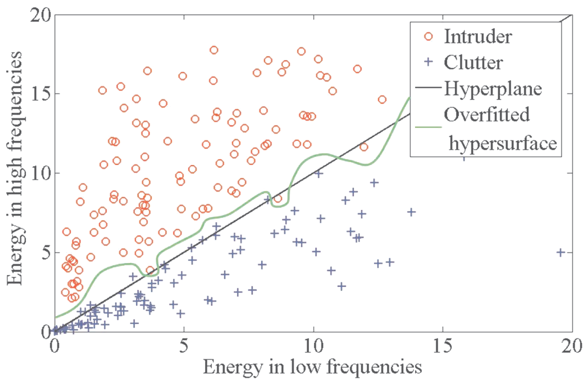
these 8 frequency "bins". The Haar signals associated with an example $N = 8$-sample transform are shown in Fig. 7.

The time-localization aspect of the HT allows reuse of components of a previously-transformed vector for the current window whenever there is overlap of the current window with the prior window.

Support Vector Machines

The SVM is a machine-learning technique used for classification which when input with two labeled sets of data (here binned vectors) returns a decision surface which tends to maximize the margin between the two data sets [13]. Under linear SVM, the decision surface is chosen by SVM to be the hyperplane in the input space that maximizes the margin, i.e., the one that maximizes the distance between the hyperplane and the input data sets. In our case, the input space is 8 dimensional. The optimal hyperplane is typically found in practice by reformulating it as a quadratic programming optimization problem which can be efficiently solved. If the input data in linear SVM are not separable by a hyperplane, we allow training errors to occur. The tradeoff between the margin and the training errors can be controlled by a parameter C in SVM. The larger the value of C, the lesser the number of training errors leading to a smaller margin. An alternative means of handling training errors is to use a nonlinear separating surface. In quadratic SVM, the optimal decision surface is chosen to be the hyperplane in a larger dimensional space which maximizes the margin from points in the larger dimensional space to which points in the data set are mapped. Each coordinate in the larger dimensional space is associated to a unique monomial of degree 1 or 2 in the variables attached to the input space. Thus the larger space is of dimension $\binom{8}{2} + 8 + 8 = 44$. One caveat is that there is a risk of over-fitting the data with a high degree separating surface that will perfectly separate the training data but is likely to fail on any new data (see Fig. 8).

Figure 8: Illustration of overfitting SVM with higher degree polynomials in $2D$.



Computational complexity

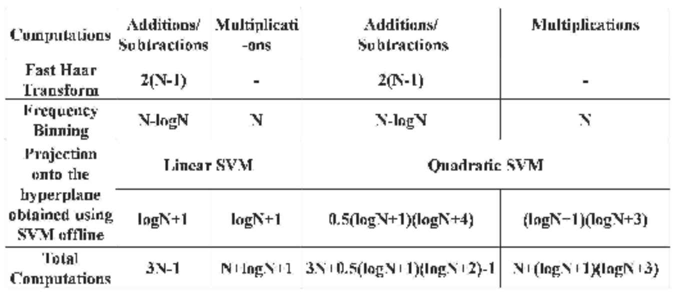
Since training of the SVM was done offline, in estimating the computational complexity, we consider only the computations carried out online in the mote on the incoming data. We consider linear SVM classifiers initially and quadratic SVM classifiers subsequently. SVM classification involves calculating $\gamma = \mathbf{w}^T\mathbf{x} + b$ where $\mathbf{w}$ is the normal to the hyperplane, b represents the affine shift of the hyperplane and $\mathbf{x}$ is the binned vector. Thus γ is proportional to the distance of $\mathbf{x}$ to the hyperplane. The number of computations required for the online part of the algorithm for both linear and quadratic SVM appears in Fig. 9. Computations are in the order of input size 'N'.

Training and testing

The data used for training SVM was collected in a laboratory (i.e., clutter-free) environment (see Fig. 10) in the case of intruder and across 20 outdoor locations on the forested campus of the Indian Institute of Science (IISc) (see Fig. 11) in the case of clutter. The training data set for intruder includes data collected after making a human walk along different straight lines oriented in a variety of ways with respect to the sensor. The clutter data was accumulated over the 6-month period from October 2008 to March 2009. This would involve setting down many nodes on the ground in selected clutter-prone areas and letting them observe and record data.

The data streams collected were partitioned into two sets, one used for training, the other for testing. The data streams from some of the more challenging environments (for example, a sensor

Figure 9: Computational complexity of the algorithm.



Computations	Additions/ Subtractions	Multiplicati -ons	Additions/ Subtractions	Multiplications
Fast Haar Transform	2(N-1)	-	2(N-1)	-
Frequency Binning	N-logN	N	N-logN	N
Projection onto the hyperplane obtained using SVM offline	Linear SVM		Quadratic SVM	
	logN+1	logN+1	0.5(logN+1)(logN+4)	(logN−1)(logN+3)
Total Computations	3N-1	N+logN+1	3N+0.5(logN+1)(logN+2)-1	N+(logN+1)(logN+3)

Figure 10: The indoor location used for accumulating intruder data.

False Alarm Rate: The probability that clutter or noise are misclassified as an intrusion.

Figure 11: A location in IISc where a part of clutter data was accumulated.

Miss Probability: The probability that an intrusion is misclassified as noise or clutter.

placed in the close proximity of large fern, or an intruder moving with high velocity) were used for training. From the data streams to be used for training, a total of 224 blocks were extracted, 112 each representing intruder and clutter. Each block is a vector containing 128 consecutive time samples. The offline training of SVM was done using LIBSVM [17] in MATLAB. Linear SVM, with C set to a high value, was used. The training performance recorded $7/112 = 6.3\%$ misses and $4/112 = 3.6\%$ false alarms. It should be noted that the training set deliberately includes some data that was hard to classify. Not surprisingly, testing performance was significantly better than training performance. The testing performance recorded $3/500 = 0.5\%$ misses and $2000/160000 = 1.25\%$ false alarms. Some representative samples appear in Fig. 12. Fig. 12 shows on the left, four intrusions when the intruder sprinted four times in front of the sensor over a period of 70 s, all of which are successfully detected by the algorithm. To the right, Fig. 12 shows clutter data collected over a period of 80 s, the clutter was successfully rejected.

Limitations and future work

As mentioned, the data reported above was for the period October 2008 to March 2009. However, when we carried out testing around noon on a sunny day in April 2009, the height of summer in Bangalore, we observed a significantly larger false alarm rate. Fig. 13 shows a sample waveform recorded in this period. When the mid-summer noon's data were also included in the training set, linear SVM recorded a training performance of $60/275 = 21.8\%$ misses and $22/275 = 8\%$ false alarms. Testing performance recorded $30/300 = 10\%$ misses and $6100/100000 = 6.1\%$ false alarms. Replacing the linear SVM with a quadratic SVM, improved the record on training data to $47/275 = 17\%$ misses and $15/275 = 5.5\%$ false alarms. The improvement with regard to testing data was far more pronounced. The use of multiple sensors may permit reduction of the false alarm rate since the false alarms were often caused by the movement of vegetation in close proximity to a sensor. An alternative approach would be to detect intruders based on a better understanding of the signature waveform of the intruder and clutter.

Field testing

Field testing was conducted on the lawns of Electrical Communication Engineering (ECE) Department in the IISc campus. The initial decision was to deploy the sensor nodes in the form of a linear array with inter-node spacing chosen to maximize the area covered by a single node while ensuring that every point in the sensing range was covered by at least 3 nodes. The idea here was that the sensing nodes would serve as a "*wireless trip wire*" (see Fig. 14). Larger areas can be covered by interlacing many such wireless trip wires. It was found that a single linear array would on occasion, fail to detect intruder moving at high-speeds, possibly because at

Figure 12: Linear SVM: Intrusion detected on the left; clutter rejected to the right.

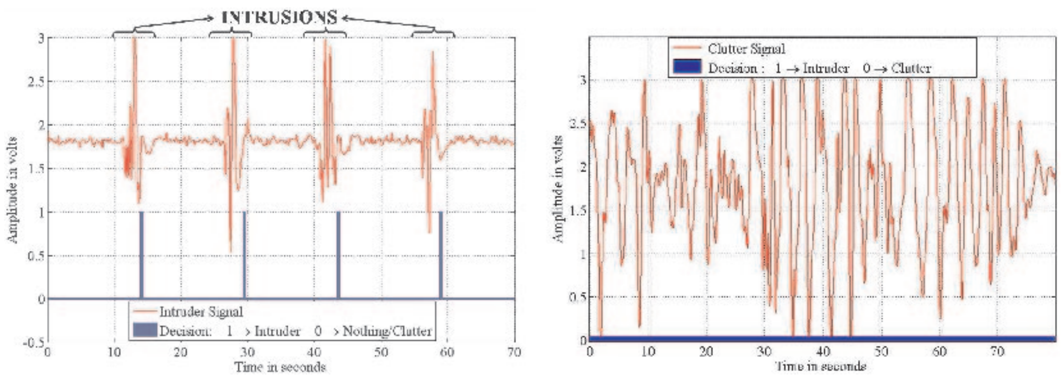


Figure 13: Quadratic SVM on summer clutter data.

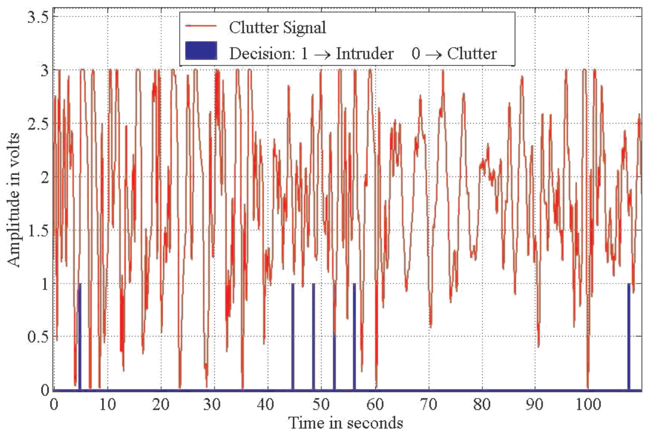
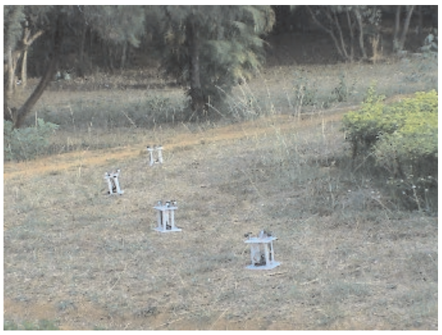


Figure 14: Three sensors platform and the wireless trip-wire.



Scalability: The property of an algorithm that enables its easy adaptation to situations with a large number of nodes.

ZigBee protocol: A simplified communication protocol based on the IEEE 802.15.4 standard for low data rate personal area networks.

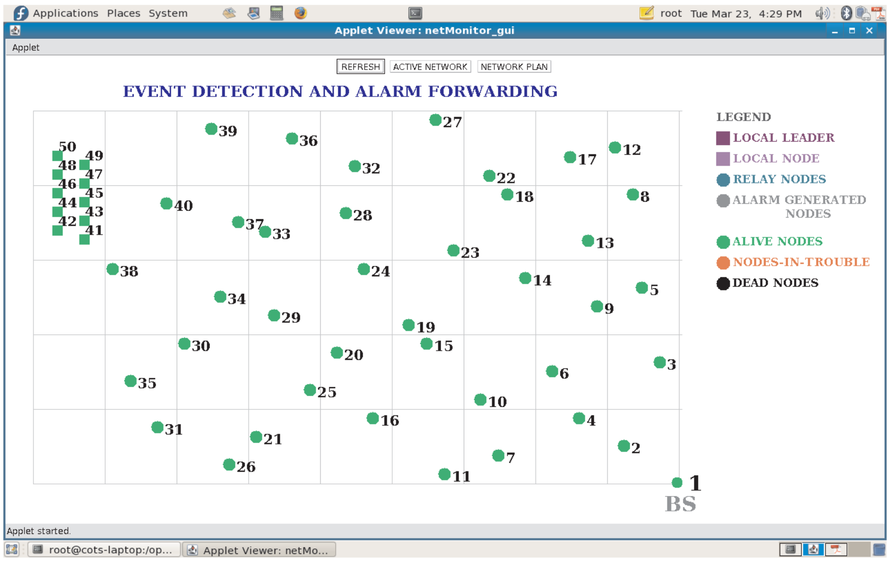
high speeds, the intruder was in the field of view of a sensor for only a very short duration. We therefore decided to create a double array comprising of two identical, linear and parallel arrays spaced apart by 5 m (the maximum sensing radius is 6 m). Decisions were made locally as follows: If a node detected an intruder in its vicinity using the HT-cum-SVM based algorithm outlined earlier, it would broadcast its local detection (via the ZigBee protocol available on TelosB motes) to all of its neighbours. A node was permitted to declare a *confirmed* detection if, in addition to making a local detection, it also received news of local detection from any other node within a distance of twice the sensing range of each sensor. The confirmed detection was then relayed back to the base station using an appropriately designed network routing algorithm. At the base station, a graphical user interface (GUI) would display the information regarding the nodes that detected and the route of the confirmed detection. This algorithm is scalable as the detection of an intruder results from the consensus of a few neighboring nodes. When tested over a period of several hours across the week, the network performed flawlessly by detecting every intrusion at speeds ranging from that of a slow crawl to a run at 5 m/sec. There were also no false alarms in the period over which testing was conducted.

3. Alarm forwarding over a wireless mesh network

3.1. *Node placement*

In SmartDetect, the nodes can be functionally categorized as sensor nodes and communication nodes. Whereas the sensor nodes take part in event detection, the communication nodes forward information about detected events to the base station. The sensor nodes are placed in two parallel rows so as to form a wireless trip-wire. The sensor nodes are separated by a distance based on the sensing range of a PIR sensor, which is of the order of a few meters. In our demonstration

Figure 15: A typical deployment of sensor nodes.



Self-organisation: As applied to communication networks, this is a means by which a topology or some sort of order emerges in the network. There is usually limited or no central intervention, and all actions are based on myopic or distributed decision making by participating agents.

Downstream nodes: The set of nodes that are closer to the destination and reachable from a given node.

setup communication nodes are placed in a semi-planned manner in which the area of deployment is tessellated into square cells, and the nodes are placed within these cells randomly. This approach emulates the practical situation that due to the presence of obstructions such as trees and taboo regions such as ditches, it may not be possible to place nodes in a perfect grid. Work is currently under way to develop methodologies for the design of wireless mesh networks (which involves node placement, and topology design) so that predictable performance can be achieved in the delivery of information over wireless sensor networks (see [12]).

After the deployment, the base station disseminates location information to all the nodes. This information is later used for routing and localising the events. In the present implementation, the nodes are scheduled to stay awake all the time. Work on a sleep–wake cycling system is under way at the time of writing this paper. The network can be monitored from the base-station using a user-friendly graphical user interface (GUI). Figure 15 shows a frame-grab from the GUI depicting 50 nodes with communication nodes (filled circles), sensor nodes (filled squares on the top-left), and the base station (BS).

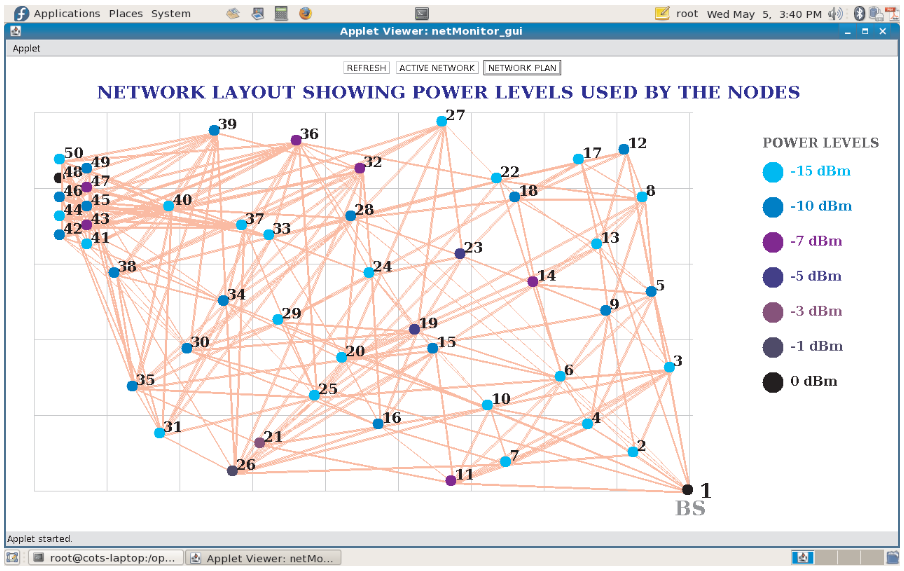
3.2. *Network self-organisation*

Self-organisation is an essential step in the formation of an ad hoc wireless mesh network. In SmartDetect, self-organisation is done via a choice of transmit power level at each node. Each node chooses the minimum power level so as to have a certain minimum number of "good" neighbor nodes that are closer to BS; these nodes will also be called *downstream nodes.* By "good" we mean that these nodes can be reached with high packet reception probability. In our implementation, we require each node to have at least 2 downstream neighbours.

In [52], the authors provide a distributed algorithm for constructing an approximate minimum spanning tree called a Nearest Neighbor Tree (NNT). The NNT algorithm bypasses the costly step of cycle detection completely by a very simple idea: each node chooses a unique rank, a quantity from a totally ordered set, and a node connects to the nearest node of higher rank. This immediately precludes cycles, and the only information that needs to be exchanged is the rank. The technique we used is an incremental neighborhood exploration similar to [52]. The Euclidean distance from the BS is used as the rank by the nodes in the self-organisation process. The nodes closer to the BS have smaller ranks than the ones farther away. The BS has the least rank.

The self-organisation process works as follows. Each node in the network does this exploration one after the other in a time ordered fashion, starting from the base station. The base station starts the process by broadcasting request packets with the least power. On the reception of a *request* packet, the receiving nodes respond to the sender by sending back an *available* packet using the same power level that is used by the sender of the request packet. The idea of sending the available packet using the same power level used by the sender is to inform the

Figure 16: A self-organised sensor network with 50 nodes.



Geographic greedy forwarding: A routing protocol in which packets are forwarded to a locally optimal next-hop node with positive spatial progress towards the destination.

sender that they can communicate at that particular power level. On reception of the available packet, the sender of the request packet stores the relevant information (for example, the number of available packets, the node ID, the power level, etc.).

After sending out a predefined number of request packets, the node checks for good neighbours in the stored information. If a sufficient number of such nodes is not discovered, the node repeats this entire process with an increased power. This process ensures that by the end of the self-organisation process, every communication node will have multiple paths to the BS, while using minimum power.

In order to identify downstream neighbours, each node compares its rank with that of its neighbours. Figure 16 shows the result after self-organisation for a 50 node network. Nodes are shown in different colours to indicate the different power levels according to the table on the right side. After due consideration to the possibility of node failure and battery depletion, this self-organisation process can be scheduled at regular intervals to maintain connectivity to the base station.

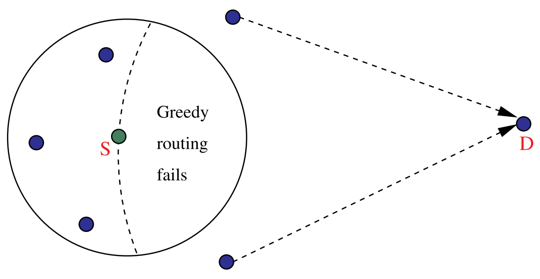
3.3. *Alarm forwarding*

Alarm forwarding in the network involves routing of the alarm packets from the originating node to the sink. For such problems, *geographic routing* [39] is a popular protocol for packet delivery. Geographic routing exploits the geographic information instead of topological connectivity information to route packets to the destination. Geographic routing protocols require only one-hop geographic information of neighboring nodes. The highly localized operation and the stateless feature of geographic routing make it simple and extremely scalable. In geographic greedy routing, packets are forwarded to a locally optimal next-hop node with a positive progress towards the destination node. Such a protocol requires a node with a packet to be aware of its geographical location, and those of its neighbours. This next hop node in the direction of the destination is called a *relay or greedy node*.

Battery-operated networks adopt radio duty-cycling [15,77], a MAC layer technique, to improve the lifetime of the network. Here each node periodically cycles between an awake state (radio ON) and a sleep state (radio OFF). This results in time varying connectivity among nodes and affects the maintenance of one-hop neighbours' geographic information required for greedy routing. Further, the radio duty-cycling leads to the optimal relay node selection problem where there is a trade-off between the delay in relay node selection and the progress made towards the sink. Thus it is imperative to develop a geographic greedy routing protocol that provides a framework for incorporating the optimal relay selection algorithm without the maintenance of one-hop neighbor information.

We propose Geography-aware MAC (GeoMac), a transmitter initiated geographic greedy routing protocol. Here a node with a packet to forward initiates handshakes by broadcasting probes at regular intervals in order to determine greedy nodes.

Figure 17: The failure of geographic greedy routing. *S* is a *stuck* node.



Pseudo-events: Virtual events (such as a fire-drill), created across space and time, that monitor the health of the network.

TinyOS: A component-based event-driven open source operating system for motes.

The probes contain the forwarding metric (distances or hop counts) and the ID of the node broadcasting it. Nodes closer to the destination respond with a probe ACK which contains its ID and forwarding metric. An optimal timer starts on the reception of the first probe ACK. All the received probe ACKs within the optimal time period are queued. The sender now unicasts the packet to the relay node closest to the destination. The value of the optimal timer determines the relay node to which the packet will be forwarded.

Geographic greedy forwarding may fail when all the neighboring nodes of a sender are farther away from the destination node than the sender itself. Figure 17 presents the local minimum condition (or a communication void [24]) where node *S* does not have any greedy relay nodes in its communication neighborhood. *S* fails to locate a next hop node in its neighborhood that has a positive geographic progress towards the destination node. Although a dense deployment of nodes can reduce the likelihood of occurrence of a void in the network, packets may encounter voids with nonzero probability and cannot reach the sink. This is undesirable for WSNs deployed for applications such as intrusion detection. Hence, there is a need to develop fault tolerant techniques that makes the network resilient to communication voids.

Solutions such as face routing [39], convex hull routing [47] and link reversal routing [25] were proposed in the literature to pull the network out of a local minimum condition. All these algorithms require knowledge of one-hop neighbours. Maintenance of one-hop neighbor information in duty-cycled sensors involves several message exchanges between nodes and its one-hop neighbours, associated access issues and collision resolution mechanisms. This can be both time and energy consuming.

In geographic greedy routing, packets are always forwarded to nodes that make positive progress towards the sink. Clearly, the resulting routing graph is a directed acyclic graph (DAG). A DAG is said to be destination-oriented when there is a directed path in the DAG from any node to the sink [25]. A DAG is destination-disoriented if there exists a node other than the sink that has no outgoing links. Such a node is said to be *stuck*. A destination-oriented network under geographic greedy routing may be rendered destination-disoriented in the presence of communication voids. Gafni and Bertsekas in [25] propose two general classes of *link reversal algorithms* for solving the above problem in a *neighbor aware* fashion. The link reversals are achieved by distributed relabeling. We provided two *neighbour oblivious* algorithms that stay within the framework of [25], and, thus have all the desirable properties of the algorithms reported therein. Our neighbour oblivious algorithms do not need to store neighbour information, but learns it as and when needed.

In a sleep-wake cycling network, however, carrying out the void removal algorithm at the time of an alarm incurs large alarm forwarding delays [16]. We proposed the use of *pseudo-events* to maintain the network in a destination-oriented state before the onset of real events. Pseudo-events create virtual events distributed across space and time. This initiates link reversal at stuck nodes while relaying these pseudo-alarm packets to the sink. Our proposed maintenance technique is likely to be more energy efficient and works well even in a duty-cycled network since the forwarding protocol itself repairs the network.

4. SmartDetect implementation

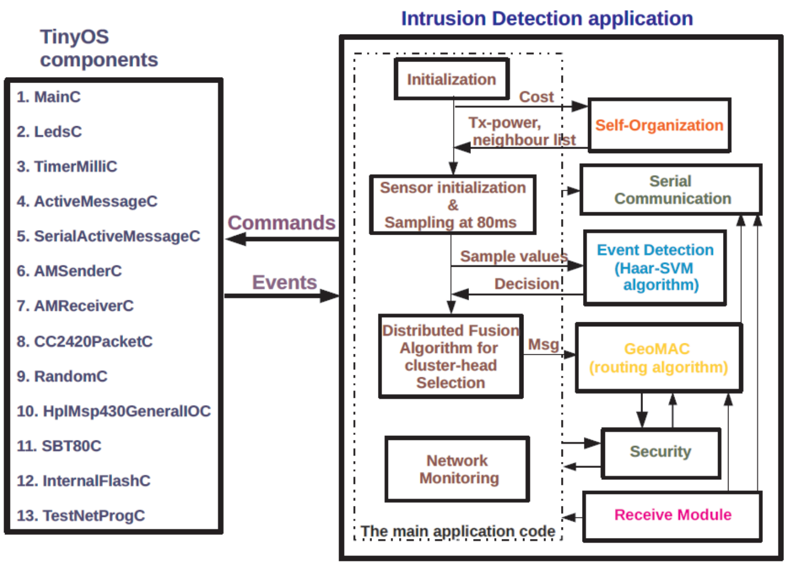
4.1. *Software architecture*

SmartDetect runs on TinyOS-2.1 operating system and the code is written in the nesC language. TinyOS is a component-based event-driven open source operating system. The application code is built using the basic building blocks provided by the TinyOS core. The TinyOS core provides components for managing timers, performing radio communication, collecting sensor samples from the analog-to-digital converter (ADC), scheduling various other tasks and so on.

Figure 18 depicts the complete software architecture of SmartDetect along with the TinyOS components it uses. The application code is modularized and provides flexibility to replace existing modules with others of the same functionality.

The application flow is as follows. The functions of the network start with the initialization of nodes in the main application code. The base station sends location information and other parameters necessary for network operations to all the nodes.

Figure 18: SmartDetect Software Architecture.

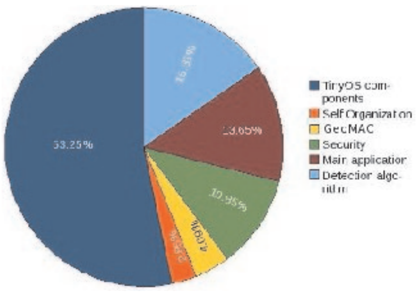


The nodes start the self-organisation process once they receive their coordinates. At each node, the output of this process is the neighbor list and a power level (just sufficient for communication directed towards the base station). This power level will be set by the node and used for further communication.

Once the self-organisation process completes, the sensors are initialized and they are sampled at regular intervals of 80 ms. The sample values obtained are fed to the detection module which performs the first level of processing by using the Haar-SVM algorithm (see Section 2.2.1). The result of this algorithm is the decision of whether there was an intruder or not. Based on this result, distributed algorithms for event declaration are run in the main application code.

Over-the-air programming: A method by which software upgrades are received wirelessly by the motes.

Figure 19: TelosB program memory utilisation in the SmartDetect implementation.



The packets that are to be sent to the base station are handed-over to the Geo-MAC module which forwards them to the base station. At each node along the way, the security module is responsible for encrypting and decrypting the messages while forwarding. Only successfully decrypted messages are passed on to the main and routing modules for further processing. Once the packets reach the base station, they are handed over to the host over a wireline serial communication interface.

The information received by the base station regarding the event is displayed on a Java based GUI. We have recently incorporated a network monitoring module which provides information about the health of the network. The information obtained by querying the nodes is displayed on the GUI. For troubleshooting purposes, we have also introduced mechanisms for requesting and getting state information pertaining to a software module within a node.

4.2. *Implementation challenges*

The program memory available on the TelosB mote is only 48 KBytes. This limited memory has posed repeated challenges during the development of SmartDetect. The problem is aggravated by the fact that half of this memory is used up by the TinyOS operating system, as shown in Figure 19. In order to accommodate all functionalities required by SmartDetect, we made careful design choices while implementing the various algorithms, and carefully optimized the code.

One of the interesting challenges we encountered was in the context of "over-the-air programming," a very essential feature required to update the software image running on the motes in-situ. For this, TinyOS has a built-in support known as Deluge. We found that incorporating the Deluge component required 40% of the program memory. We therefore came up with a scheme involving a two step process. We make use of two application images, an application image with Deluge alone, the other being our main SmartDetect application. We temporarily move from SmartDetect to the Deluge application during the software upgrade, and "reboot" back to SmartDetect after the upgrade.

5. Time synchronisation

Two clocks never agree all the time. Disagreements arise because oscillators that drive the clocks in motes are cheap, have varying characteristics from device to device, and are sensitive to changes in the environment, particularly temperature. But synchronisation, even if only obtained to a certain degree, can indeed be useful. First, WSNs are severely energy constrained. Sleep-wake cycling of motes enhances their life time.

Skew and offset: The parameters (slope and intercept) of a simple affine model, for relating the local time at a node with that at the reference node.

Probabilistic method of key distribution: This refers to a random assignment of a certain number of keys from a common pool.

Synchronised sleep-waking can allow a sufficiently small waking time just enough to compensate for minor drifts, yet sufficient to communicate with a neighbouring sensor during the awake time, thereby increasing life time of the sensor and the network. Second, when tracking a moving object, time-stamped data from various sensors have to be appropriately sequenced for reconstruction of the intruder's path. Synchronisation is essential to achieve this sequencing. Third, secret keys for secure communication are frequently changed to improve security. Synchronisation ensures that these key changes are coordinated.

Several algorithms for synchronisation are available. The most common method is to make a reference node broadcast a packet with its time information. If propagation delays are negligible, this broadcast enables nodes to calibrate their time line with this common reference tick, and make corrections as needed. However, this only exploits the star-like connections from the reference node to the other nodes. Links between other nodes are not exploited. Solis, Borkar, and Kumar [72] provide a different algorithm that exploits these connections. In particular, the net clock offset around any loop is zero, analogous to the well-known Kirchchoff's voltage law. Similarly, the net product of the skews is 1, yielding another conserved quantity. These ideas provide a distributed and asynchronous algorithm for clock updates. The algorithm also adapts to changes in network as long as the network is able to elect one of the nodes as a leader (reference node), i.e., it is robust to a change in the leader. A rough description of the two-stage algorithm is as follows. All sensors track their skews and offsets with respect to a chosen reference node.

(1) In the first stage a node conducts bilateral exchanges of packets containing transmit and received time stamps with each of its neighbours. This enables it to estimate its relative skew and relative offset with respect to each neighbour.

(2) In the second stage this node hears broadcast information of neighbours' skews and offsets with respect to the global reference node. An estimate of the node's offset with respect to the reference is the sum of a neighbour's offset and the computed link-level offset with this neighbour. Thus each neighbour gives an estimate of the node's offset. The updated offset is the average of all these estimates.

A similar calculation is done for skews with skews and offsets computed treating the other quantity as known. The above algorithm is robust to distributed and asynchronous updates. It can also handle changes in the reference node quite gracefully. The new reference node simply stops making updates, and the old reference node begins making updates. The converged values will now indicate offsets and skews with respect to the new reference node.

The above algorithm is a discrete-version of the so-called averaging principle that underpins several physical phenomena such as heat transfer in a medium and social phenomena such as spread of gossip in a network of individuals. Simulations indicate that the algorithm can provide tens of microseconds accuracy, with the limitation factor being the accuracy of the time-stamping mechanism itself.

6. Network security

Given that the network may need to operate in remote and potentially hostile environments, secure message exchange is a basic need.

Secure message exchange imposes several requirements, including

- Encryption: Transmissions are encrypted, so that an adversary capturing packets off the air would see only ciphertext, and not plaintext
- Authentication: A sensor node receiving a message needs to be sure that the message was transmitted by a friendly neighbour, and not a malicious adversary
- Message Integrity: Similarly, one would like to ensure that a received message has not been tampered with, and its contents have not been modified in any way.

Public-key or asymmetric cryptography and private-key or symmetric cryptography are the two choices available for implementing a secure system. Public-key cryptography imposes significantly higher computational requirements. Keeping in mind the limitations of the off-the-shelf sensor node platforms to be used in the prototype system, the option of private-key cryptography was selected.

6.1. *New key distribution algorithms*

Keys can be provided to sensor nodes *before deployment*, or they can be generated by nodes *after deployment* in the field. In the following paragraphs, we provide an overview of a few algorithms of each type that have been developed in the course of the project.

When keys are provided before deployment, there are two strategies that can be followed. In the *probabilistic* method, a random selection of k keys (say) is made from a pool of P keys, and assigned to a sensor. In this way, a random subset is chosen independently for each sensor. Thus, after deployment in the field, two nodes wishing to communicate may or may not have keys in common.

If they share one or more keys, then they are able to communicate securely. If they do not share keys, then the nodes need to execute some algorithm to obtain shared keys.

In [29], we propose the "Friends"-based algorithm to generate common keys between a source S and a destination D. Essentially, S initiates a search for nodes X that have common keys with D; these are the *friends* of S. Each X sends a *part* of a common key back to S. S constructs a full key by a random choice from the part-keys received, and informs D about the identities of those X whose part-keys were used. This enables D to construct the same full key that S has, and henceforth, the two can communicate securely.

Friends of a given node are those nodes that share a common key with the given node.

Evaluation of the proposed algorithm considers the *feasibility* of finding enough friends for the scheme to work, as well as a measure for quantifying how *secure* the generated key is — the probability that an adversary is able to recover the generated key in L attempts. A third metric used is the *energy overhead* of the method [30]. Analytical expressions are obtained for each metric, and comparisons with a related scheme in the literature [50] yield encouraging results.

The second method of providing keys before deployment is a *deterministic* method, in which keys are allocated to nodes explicitly. In [28], we propose a method in which a pool of p^2 keys (p: prime) is arranged in $p \times p$ grid, and a sensor node corresponds to a polynomial over the residue field $\{0, 1, \ldots, (p-1)\}$. The polynomial passes through points in the grid, and these keys are allocated to the node. The idea is a generalization of [38].

Evaluation of the algorithm considers the *connectivity* and *resilience* metrics. Connectivity refers to the probability that two randomly chosen nodes have at least one common key between them. Resilience indicates the fraction of keyed links that are compromised when c nodes (say) are captured by the adversary. Analysis yields formulae for the metrics, and suggests design rules in terms of how to choose p in order to achieve desired values of connectivity and resilience. A particular case of the deterministic algorithm is the *full connectivity* scheme, in which connectivity is guaranteed. Our work in [28] includes an algorithm for assigning keys such that full connectivity is ensured.

Next, we consider algorithms for key generation *after* nodes have been deployed. The topic has received a lot of attention in the literature, and a recent trend is to examine if knowing the *expected locations* of nodes helps in improving connectivity and security. In [2], we propose an algorithm for generating keys *in situ*, after nodes have been deployed. The algorithm is initiated by a few nodes called "tagged nodes," that are programmed to transmit a few broadcast packets after deployment. Essentially, each tagged node can be thought of as a leader with its own pool of keys, and different tagged nodes have disjoint key pools. Normal nodes — which are exactly the same as tagged nodes except that they do not send initial broadcasts — receiving messages from leaders associate themselves to a chosen leader's group (some nodes choose more than one leader). When a few nodes are compromised, the revealed keys affect communication among relatively few nodes; this is because sensors associated with distinct tagged nodes use distinct key pools. The idea is to restrict the impact of loss of keys to a limited area, rather than the complete network. Simulations indicate that both connectivity and security improve compared to schemes that leverage expected node locations; this is interesting, because the proposed scheme does not use knowledge of node locations in any way.

6.2. *Implementation on off-the-shelf devices*

Implementing security on commercial off-the-shelf (COTS) sensor motes posed its own set of challenges. As indicated earlier, the TinyOS operating system and other essential system-level code occupied a substantial part of the memory available in the TelosB motes that were used. As a result of the memory constraints, only a simple and lean security implementation could be attempted.

For encryption, the 128-bit AES algorithm was used. AES is a block cipher which is obtained as a special case of the Rijndael algorithm. We ported an existing C code for the algorithm into nesC for use in the motes.

For authentication and message integrity check, the CBC-MAC framework was used. In this, a block cipher is used to calculate the message authentication code (MAC). Our implementation used AES itself as the block cipher.

Before deployment in the field, each sensor node was provided with the same secret key. Each message was encrypted using the secret key and the AES algorithm. Authentication and message integrity tags were calculated and transmitted along with the message bits, using the AES-CBC-MAC framework.

Using a single secret key throughout the network is risky, because a compromised node may end up revealing the secret key and thereby jeopardizing communication over the entire network. To address this problem, periodic rekeying was implemented. The basic idea is that at regular intervals, each node generates a new key for itself by executing a hash function with the old key as the argument. The Grindahl hash function was used for this purpose.

Loose time synchronization among the nodes was assumed; this ensured that nodes executed the hash function at roughly the same intervals. However, to allow for occasional mismatches, a window of three keys — past, present and future — was maintained by each node.

A basic replay attack protection feature was incorporated into the implementation. This relied on including a timestamp with each message. A replayed message would be caught by the old timestamp that it carried.

The implemented security features were demonstrated on several occasions. An outsider attacker not possessing the secret key and trying to communicate is caught immediately by genuine sensors nodes in its vicinity, and an alarm message is sent to the Base Station. Apart from periodic rekeying, the Base Station can command all nodes to switch to a new key at any time; this feature can be utilized to thwart a suspected adversary. Our current work is aimed at tackling the harder problem of an "inside adversary," who has access to the secret keys and behaves in a malicious manner.

7. Event detection: other statistical formulations

The simplest model for intrusion is one where sensor observations are modeled as being sampled from a certain distribution prior to the intrusion, and as being sampled from a certain other distribution after the intrusion. The observations at a particular time instant are naturally spatially correlated, i.e., correlated across sensors, because they are observations at the same sampling instant of the same phenomenon (intruded or unintruded state). The goal is to detect the disruption (intrusion) as soon as it occurs, but not before. Given the stochastic nature of the system, one tries to minimise the expected delay for detection after the intrusion, subject to a maximum tolerable false alarm rate.

Bayesian inference problems are those in which the decision maker is supplied with prior probabilistic information about the hypotheses.

Beamforming: A means by which signals transmitted by different antennas phase-align at the receiver.

Markov decision process (MDP): Another term for a controlled Markov chain, which is a Markov chain in which the transition probabilities depend on the current state and the current action.

7.1. *Bayesian event detection in a small network*

The classical Bayesian sequential change detection problem [70] is formulated as follows. There is a random (discrete) time Γ_1 at which the change occurs. The *prior* information includes the distribution of Γ_1. All the sensors take periodic samples of what they are designed to observe: temperature, strain, etc. In the small network setting, the change affects all the sensors in the same way, so that conditioned on $\Gamma_1 = t$ the observations at the sensors up to $t-1$ are assumed to be independent and identically distributed over time and across the sensors, with probability distribution $F_0(\cdot)$. Similarly, the observations are independent over time and sensors from t onwards but with a different probability distribution, $F_1(\cdot)$. At each time k, the detection procedure makes a decision as to whether the change has occurred at or before time k or to continue making measurements at time $k+1$. The optimal change detection procedure raises an alarm at time τ (which depends only on the observations made until then) such that the mean detection delay $\mathsf{E}\left[(\tau-\Gamma_1)^+\right]$ is minimum over the class of all stopping times τ whose probability of false alarm $\mathsf{P}_{\mathsf{FA}} = \mathsf{P}\{\tau < \Gamma_1\}$ does not exceed α, a suitably chosen small value, such as 0.01.

7.1.1. *Physical layer fusion*

Typically, the observations made by the sensors are quantised, coded, and then sent individually over the air to an information fusion and decision unit. This assumes an ad hoc separation of quantisation and transmission over the wireless channel. Moreover, each of these nodes contends for the common channel. In the classical approach, after receiving all observations, the fusion centre fuses these observations (for e.g., addition of log likelihoods of individual observations) because they are spatially correlated. Given this eventual additive processing at the fusion centre, can superposition of electromagnetic waves provide this fusion? Such an approach will not only do away with the need for channel contention, but also perform a joint source and channel coding, a form of analogue coding.

In our recent work [79], we considered a "star topology" of a fusion centre surrounded by several nodes with links from the nodes to the fusion centre. The sensors first calculate the log-likelihood ratio of their observations, where the ratio is of likelihoods under the changed distribution over the likelihood under the no-change distribution. The sensor transmitter then amplitude modulates a standard waveform with this value. It also applies a phase correction based on channel knowledge from the base station to the node assuming that a TDD system is used and channel *reciprocity*. Superposition of the transmitted waveforms does the rest and computes the needed statistic for decision making with the caveat that this statistic is embedded in receiver noise. This mechanism is called *distributed beamforming*, because waveforms align at the receiving end. The fusion centre then makes a decision whether to stop or to continue sampling. It can also provide feedback to enable energy savings.

The problem can be cast as a Markov decision process. The decision at each stage is whether to draw a new set of samples or stop and declare an intrusion. When the decision is to draw a new sample, the choice of parameters of the affine transmission optimise energy consumption. We

Partially observed MDP (POMDP): A type of Markov decision problem in which only a noisy version of the state is available to the decision maker.

CUSUM algorithm: A specific sequential decision algorithm that accumulates evidence in favour of the alternate hypothesis.

provide an explicit closed form solution for the control at each stage, when the prior and posterior distributions are Gaussian, but with different means.

The biggest challenge in this proposed scheme is the practicality of distributed beamforming. Clocks are required to be synchronised so that carrier phases and symbol ticks align. Engineers at the University of California, Santa Barbara have recently embarked on an effort to build a prototype distributed beamforming system [58].

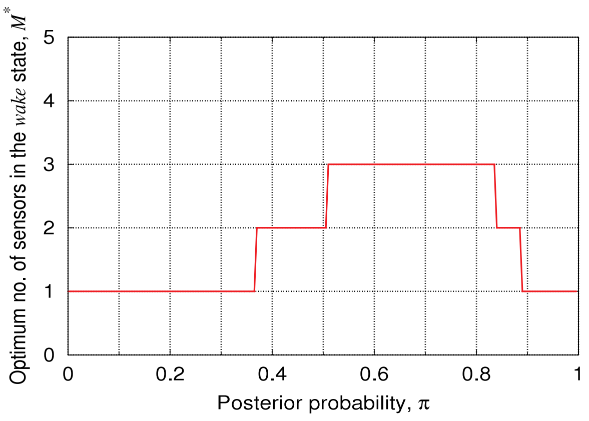
7.1.2. *Optimal sleep-wake cycling of sensor nodes*

In the classical problem [70], the only tradeoff being accounted for is that between detection delay and the probability of false alarm. In intrusion detection applications using WSNs, the energy cost (i.e., sensing + computation + communication cost) of using a sensor is a critical issue as sensors are energy-limited devices, and an intrusion is typically a rare-event. We consider a small extent network in which there are N sensor nodes. We are interested in obtaining a detection procedure which, at every time k, makes a decision as to whether the intrusion has occurred or to continue making measurements at time $k+1$ with $M_{k+1} \leq N$ sensors. Thus only these M_{k+1} sensors will expend energy to provide measurements at $k+1$, while the rest can stay in a low energy state. The energy-optimal detection procedure minimises the sum of detection delay and energy costs subject to the constraint on the false alarm probability.

The problem can be formulated as a Partially Observable Markov Decision Process (POMDP) with the information state at time k being Π_k, the posterior probability of change having occurred at or before time k given the observations [64]. As is usual for such problems, a Bellman equation can be written down. Some properties of the optimal policy can be obtained and numerical evaluation of the optimal policy can be done by value iteration.

From Figure 20 we see that when Π_k is either low or high, it is enough to keep one sensor awake, and when Π_k is close to 0.5, we need to use more sensors. This can be interpreted as follows: when the posterior probability of change is low or high, the uncertainty about the occurrence of the event is small, and when the posterior probability of change is close to 0.5, the uncertainty about the event is large. If the uncertainty is large, we need more sensors to resolve the uncertainty.

Figure 20: Optimal sensor wake-up policy (number of sensors to be used as a function of the posterior probability) for $F_0 = \text{Normal}(0,1)$, $F_1 = \text{Normal}(1,1)$, sensing cost $\lambda_s = 0.5$, and false alarm cost $\lambda_f = 100.0$ (this sets P_{FA} to 0.04).



7.1.3. *Detection over a network, and transient change detection*

In practice, the communication between the sensors and the fusion centre is facilitated by a wireless ad hoc network. The media access control (MAC) layer of the network introduces random delays in receiving the observations and hence, the observations may not be received in chronological order at the fusion centre. We study the sequential change detection problem in the presence of network delays in [67] and show that the optimal decision procedure requires additional knowledge of the queueing state of the system.

Also, the change detection problem considered so far assumes that the change persists for ever. However, in intrusion detection applications, the change is transient (i.e., the change disappears after a random time in the network). We formulate and study this problem in [66].

7.2. *Non-Bayesian formulations: distributed detection*

7.2.1. *The DualCUSUM Algorithm*

We have developed an algorithm, which we call DualCUSUM, which is based on the well known CUSUM algorithm [61]. In DualCUSUM a local detection is made at each node by using the CUSUM algorithm, and the local decisions at the nodes are fused at the fusion centre using another CUSUM. The following are some features of this algorithm. Details can be found in the related publications [9,37].

The local CUSUM threshold at each node is chosen appropriately to satisfy the overall system false alarm probability target. Each sensor node transmits data (in fact just a bit) to the fusion node only if it declares a change. As a change occurs but rarely in our intrusion application setting, in DualCUSUM a local node will need to transmit very rarely. This leads to significant saving in energy used in transmission. Node i transmits its bit at power P_i, which is a design parameter. All nodes can transmit simultaneously. This saves MAC delays and leads to overall reduction in detection delay at the fusion node. For this *physical layer fusion* to be possible one needs tight time synchronization of different nodes (see Section 7.1.1).

Physical layer fusion: The means by which the signal processing step of accumulation of evidence at the receiver is performed by the superposition of electromagnetic signals.

Because of receiver noise, the fusion node does not know how many sensor nodes are transmitting in a slot. Thus, even if P_i and the sensor noise variances are known to the fusion node, it cannot compute the exact log-likelihood ratio needed for CUSUM. Thus, in DualCUSUM, at the fusion node, we use an appropriately designed density for the "alternate" hypothesis [9,37].

The overall performance (the probability of false alarm and mean delay) of DualCUSUM depends on various parameters. We obtained the performance of the algorithm for a given set of parameters. We then found the optimum parameters to obtain the best performance. CUSUM requires knowledge of the node transmit powers, P_i, the receiver noise variance, and the channel gains. Typically, these will not be available. In such a scenario one can use a nonparametric CUSUM or generalized likelihood ratio instead of the likelihood ratio. These variations of DualCUSUM were also studied (see [37]).

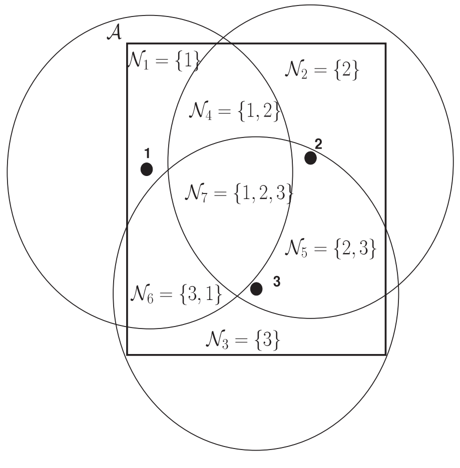
7.2.2. *Detection and localisation of an event in a large WSN*

We consider a *large extent* WSN, where an event influences the observations of sensors only in the vicinity of where it occurs. Thus, in addition to event detection, we are also interested in identifying the location of the event. As we consider a large extent WSN, a centralised solution where a fusion center makes a decision at each time k after receiving the observations from all sensors, is *not* desirable. We propose distributed event detection and location identification procedures based on CUSUM [61] and analyse their optimality properties.

An event is modeled as occurring at a point in a two-dimensional region of interest. Here we discuss the simple *Boolean model* where the post-change distribution at any sensor within a radius of r_s from the event is $F_1(\cdot)$ and that for any sensor node outside this disc is $F_0(\cdot)$. A more general sensing model is considered in [65].

Figure 21 depicts the situation for the Boolean model. We partition the region of interest $\mathcal{A}$ in such a way that each sub-region $\mathcal{A}_i$ in the partition is covered by a unique set of sensors $\mathcal{N}_i$ (see Figure 21). Thus, we have a hypothesis testing problem for identifying the location of the event where the hypothesis $\mathbf{H}_i$ corresponds to the set of sensors $\mathcal{N}_i$ and the hypothesis $\mathbf{H}_0$ corresponds to *change not having occurred* so far.

Figure 21: An example of partitioning of $\mathcal{A}$ in a wide-area WSN. The three sensor nodes divide the overall region into regions $\mathcal{A}_1, \ldots, \mathcal{A}_7$ such that region $\mathcal{A}_i$ is covered by a unique set of sensors $\mathcal{N}_i$.


$\mathcal{A}$
$\mathcal{N}_1 = \{1\}$
$\mathcal{N}_2 = \{2\}$
$\mathcal{N}_4 = \{1,2\}$
1
2
$\mathcal{N}_7 = \{1,2,3\}$
$\mathcal{N}_5 = \{2,3\}$
3
$\mathcal{N}_6 = \{3,1\}$
$\mathcal{N}_3 = \{3\}$

We study distributed procedures that detect and locate an event (to any of the $\mathcal{A}_i$s) subject to a false alarm and false isolation constraint. The *false alarm constraint* considered is the average time to false alarm at the region $\mathcal{A}_i$ (TFA_i) defined as the average number of samples taken under the hypothesis $\mathbf{H}_0$ to raise a global alarm and declare the hypothesis $\mathbf{H}_i$. The *false isolation constraint* considered is the average time to false isolation TFI_{ij} defined as the average number of samples taken to raise a global alarm and declare the hypothesis $\mathbf{H}_j$ when the event has not occurred in $\mathcal{A}_j$ but has occurred in $\mathcal{A}_i$. The supremum average detection delay of a procedure τ, $\mathsf{SADD}(\tau)$ is defined as the worst-case average number of samples taken under any hypothesis $\mathbf{H}_i$, $i = 1, 2, \ldots, N$, to raise a global alarm.

Each node runs a local decision rule, and the global decision rule is to stop and raise an alarm as soon as all the sensors in any set of sensors $\mathcal{N}_i$ have a local decision 1, and we isolate the location of the event to the region $\mathcal{A}_i$. We propose three local decision rules: ALL (see [54]), MAX (see [74]), and HALL (hysteresis modified ALL), all of which are based on running the CUSUM procedure at each node.

We show in [65] that the procedures ALL and HALL are asymptotically minimax delay (SADD) optimal as the mean time between false alarms (and false isolations) goes to infinity. We compare the performance with an optimal centralised procedure due to Nikiforov [60], and find that the distributed procedures ALL and HALL achieve almost the same performance as that of Nikiforov's centralised algorithm.

8. Development of a high-resolution vibration sensor

We present here an overview of the work done on the development of a single-axis micromachined in-plane accelerometer for use in detecting intrusion by sensing the ground vibrations. Detection of intrusion caused by human foot-steps and vehicular traffic in rough terrain requires sensitivity from a vibration sensor of the order of 10^{-6} to 10^{-3} g over a few kHz frequency range (Mazarakis and Avaritsiotis [53]; Ekimov and Sabatier [23]). Commercially available vibration sensors that can detect millionth of the acceleration due to gravity (the so-called *micro-g* accelerometers) are very expensive (e.g., about Rs. 30,000 to 1,00,000 per sensor) and hence they are not economically viable for wireless sensor networks. So, the objective of this work was to design and develop a low cost, high resolution, highly sensitive, small-sized, power-efficient vibration sensor that works over a large bandwidth. A summary of the work done during the course of this project is described here.

Accelerometer: A measurement device that behaves like a mass on a spring. The displacement of the proof mass is proportional to the external acceleration.

Noise floor: The level of all unwanted signals from within and outside the measurement system.

8.1. *Initial trials with a commercial accelerometer*

In order to assess the feasibility of human footstep detection, we conducted some trials with a commercial accelerometer (LIS3LO2AS4; ST Microelectronics (http://www.st.com/stonline/). The chip along with the evaluation board is shown in Figure 22(a). It is a three-axis accelerometer with a sensitivity of 500 mV/g. Its specification states that its noise-floor is 50 μg/rootHz. However, in our trials we could not see a resolution better than a few mg when it was tested on a shaker table (LSD shaker). This may be because of the ambient noise as well as due to the mounting. This, as stated earlier, is a concern with vibration sensors as they are sensitive to the way they are mounted. To check for consistency, several evaluation boards as well as mountings were made in-house. This is shown in Figure 22(b). Furthermore, we attached them to a metal peg firmly so that the sensor platform can be fixed into the ground as shown in Figure 23(a–b). A power supply unit and an oscilloscope were kept on the side to measure the performance. It was noticed that the sensor was able to pick up human footsteps within a distance of about 1 m away from the sensor. Shown in the right panel of Figure 23 is a clear signal of about 20 mV due to a foot-thump at a distance of 1 m. It can be noticed that there was a noise of about 3 mV. This trial confirms that footsteps can be detected. For long-range detection, we need higher resolution than what was seen in this trial.

8.2. *Initial design and system level simulation*

The work began with a focus on an in-plane capacitive accelerometer. The design of the mechanical sensor element and the capacitance-sensing circuit began simultaneously. The sensor element design assumed that a capacitance-change of one part per million could be detected by the circuit. This set the goal for designing the sensitivity of the accelerometer and the minimum detectable acceleration. The circuit design assumed that the sensor element has a base capacitance of about 1 pF.

Figure 22: A commercial accelerometer from ST Microelectronics (LIS3L02AS4) with the specifications: sensitivity of 0.5 V/g, noise-floor of 50 μg/rootHz, resonance frequency of 1.5 kHz, and a cross-axis sensitivity of about 2%. (a) Sensor chip with the in-house built evaluation board and mounting; (b) schematic of the mounting and the sensor chip.

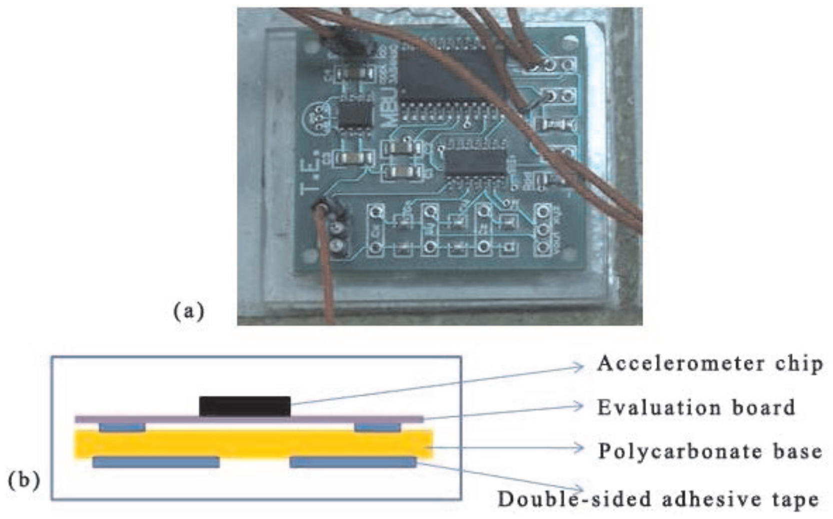
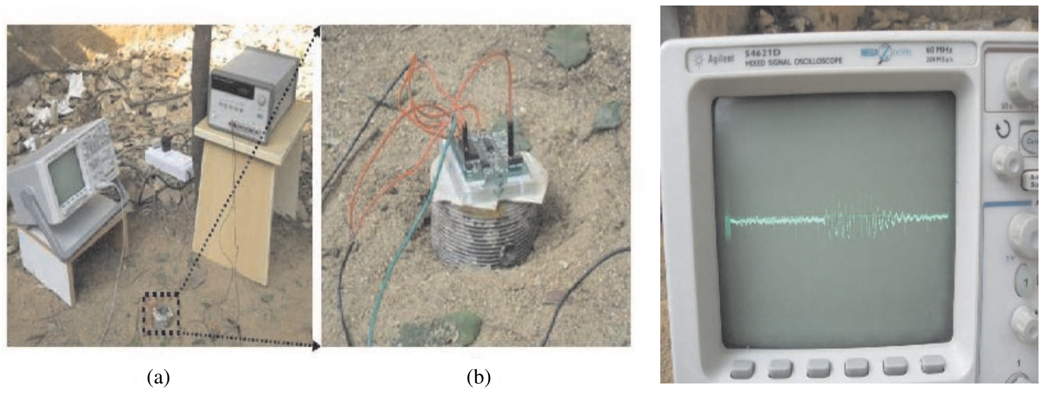


Figure 23: Testing of the accelerometer mounted in the ground. (a) The vibration sensor, the power supply unit, and the oscilloscope; (b) close-up view of the sensor; The scope picture on the right shows the captured signal for a foot-thump in the vicinity of the mounted sensor; 3 mV noise-floor and 20 mV signal.



Displacement-amplifying Compliant Mechanism (DaCM): A monolithic jointless lever that amplifies displacements using elastic deformations.

Cross-axis sensitivity: The sensitivity of the device in an axis perpendicular to the intended axis.

Bandwidth of a sensor: The number of times a reliable sensor reading can be taken per second.

An in-plane accelerometer was chosen rather than an out-of-plane accelerometer because a distinguishing feature of our approach was to enhance sensitivity through mechanical amplification and that is more amenable in the plane than the out of the plane motion. The aim here is to enhance the sensitivity and resolution of the accelerometer. This can be done by improving the capacitance-extraction circuit and signal conditioning and by amplifying the signal mechanically in the sensor element itself. With a preliminary design of the sensor element (Krishnan [42]) and the capacitance-extraction circuit (Hegde et al. [32]), it was noticed in the system-level analysis (see Figure 24) that the noise is, in general, more in the electronics than in the mechanical sensor element. Therefore, even though we can increase the sensitivity in both the mechanical and electronic parts of the sensor system, it makes more sense to increase the sensitivity in the one with less noise. This way, we do not amplify the overall noise as we amplify the signal. So, it was decided that we use mechanical amplification of the acceleration (i.e., vibration) signal to get enhanced signal-to-noise ratio.

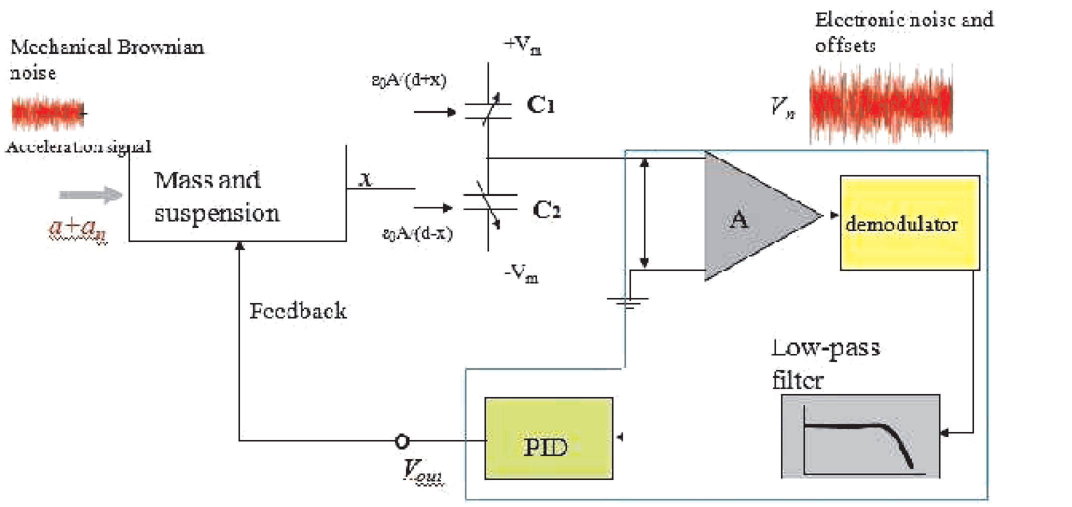
Mechanical signal amplification is achieved with a Displacement-amplifying Compliant Mechanism (DaCM) (Krishnan and Ananthasuresh [43]). Figure 25 shows a DaCM appended to a conventional accelerometer. A conventional accelerometer consists of only a proof-mass suspended by its own folded-beam suspension. The input end of the DaCM is appended to the proof-mass as shown in the top portion of Figure 25. The output end of the DaCM displaces much more than its input end and hence the capacitance-sensing comb is located at the top most part of Figure 25. A DaCM is possible for out-of-plane motion too but the required microfabrication process will be much too complicated. Hence, in-plane accelerometers are considered in this project. To demonstrate the feasibility of this concept, a meso-size (mm to cm) spring-steel accelerometer was made and was interfaced with a PCB implementation of the capacitance extraction circuit. The details of this were reported in [11].

8.3. *Micromachined chip and ASIC*

There are many trade-offs in accelerometer design: between sensitivity and bandwidth; between sensitivity and range; between main and cross-axis sensitivities, etc. Finally, the resolution, which depends on noise in the mechanical and electronics parts, also has considerable trade-off with other performance parameters (Krishnan et al.[44]). In general, thick proof-mass, low stiffness of the suspension in the main axis, closed-loop operation, and differential capacitance arrangement help. We used optimization techniques to obtain three competing designs as indicated in Table 2 (Krishnan [42]). The bandwidth of all of them was around 1 kHz. It can be noticed in Table 2 that what gives the best resolution does not have the best cross-axis sensitivity and range. These were designed with a silicon-on-insulator (SOI) process in mind. We assumed a proof-mass thickness of 250 μm and the suspension thickness of 25 μm. We kept the minimum width of the narrowest beam at 5 μm. The layout of one of the accelerometers is shown in Figure 26 with its components marked.

Due to lack of access to an SOI process that can give thick proof-mass (see Figure 28, left

Figure 24: System-level simulation of the accelerometer performed in Simulink in Matlab.



panel), we reduced the proof-mass to the same thickness as that of the suspension and DaCM (see Figure 28, centre panel). This reduced the resolution to mg rather than 10s of μg. Two designs of milli-g accelerometers using SOIMUMPs (Silicon-on-insulator Multi-user MEMS Processes) were fabricated in MEMSCAP, USA, a microsystem foundry. The process-flow of SOIMUMPs limits the thickness of the proof-mass to 25 μm [56] thus reducing the resolution and sensitivity of the device. A photograph of the SOIMUMPs die containing the accelerometer is shown in Figure 28 (right panel). The size of the die was 1 cm$\times$1 cm whereas the size of the accelerometer on the die was 6.89 mm$\times$6.07 mm. The sense-gap between the fingers of the capacitance-sensing combs was 10 μm. A close-up view of the fabricated device is shown in Figure 27. The base capacitance of the sensors was 15.5 pF when the substrate was electrically floating. It was 2.6 pF when the substrate was grounded. All the measurements were performed at a supply frequency of 1 MHz and a DC bias of 1 V with an AC level of 5 mV. The simulations showed the values to be 12.08 pF and 2.08 pF, respectively, a reasonable agreement with the measured capacitances.

Figure 25: Displacement-amplification compliant mechanism appended to an accelerometer.

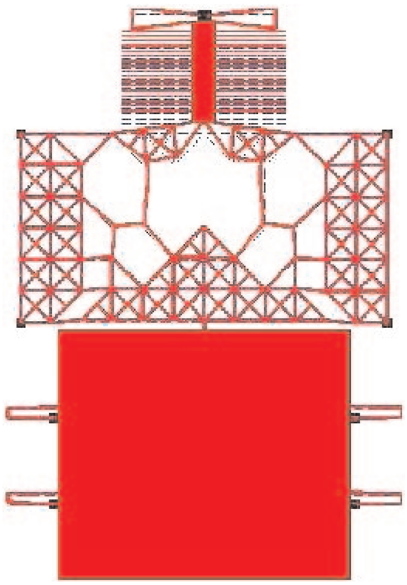


Table 2: Simulated performance parameters of three competing designs for an SOI-process

Design	Sensitivity (V/mg)	Cross-axis sensitivity (%)	Resolution (μg)	Range (mg)
1	0.125	0.030	40	40
2	0.445	0.010	20	6
3	0.070	0.005	70	70

The die-on-board packaging technique was adopted to package the SOIMUMPs die containing the milli-g accelerometer. A custom designed gold-plated PCB was used as a base onto which the SOIMUMPs die was pasted using conductive silver epoxy. In order to bond the contact pads on the accelerometer to the gold-plated bonding sites on the PCB, 1 mill Aluminum wire was used. A transparent acrylic cap was used to seal the wire bonded die hermetically to protect it from possible

Figure 26: Schematic of the mechanical sensing element with the displacement-amplifier, proof-mass suspension, suspension on the sensing side, and comb arrangement for capacitance detection and feedback stabilization.

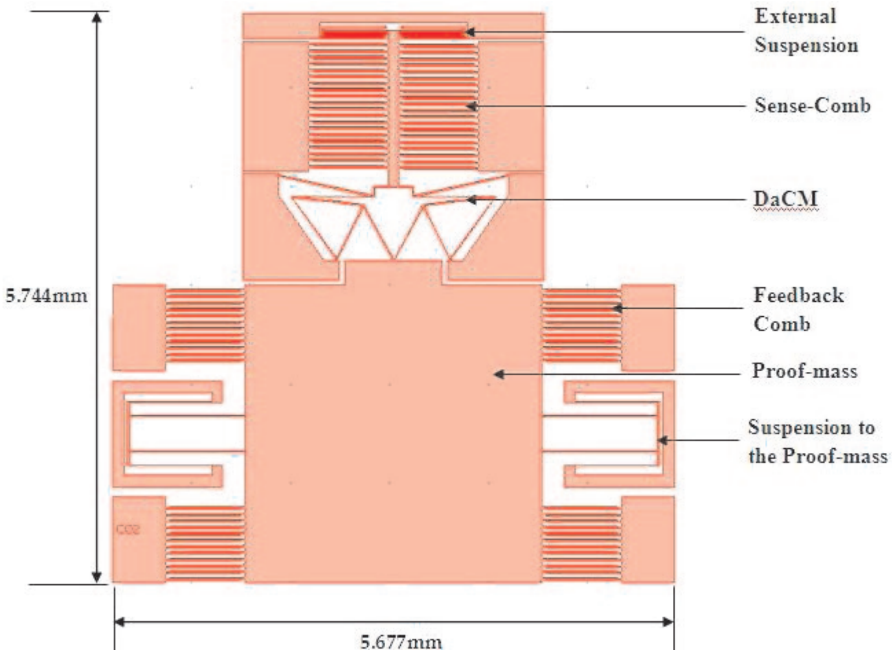


Figure 27: A close-up view of the displacement-amplifying compliant mechanism (DaCM) as fabricated in an SOIMUMPs run.

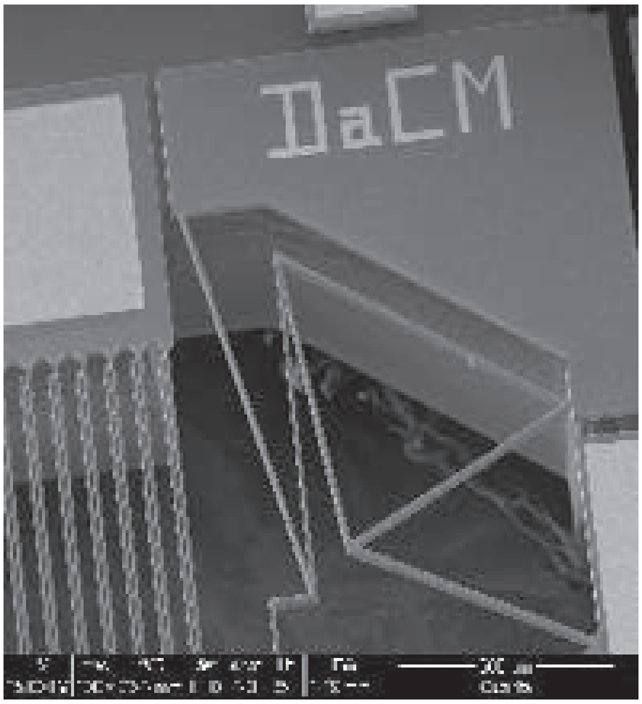


Figure 28: Left and center panels: Solid models of the high-resolution accelerometer. Right panel: The fabricated SOIMUMPs die showing its various parts.

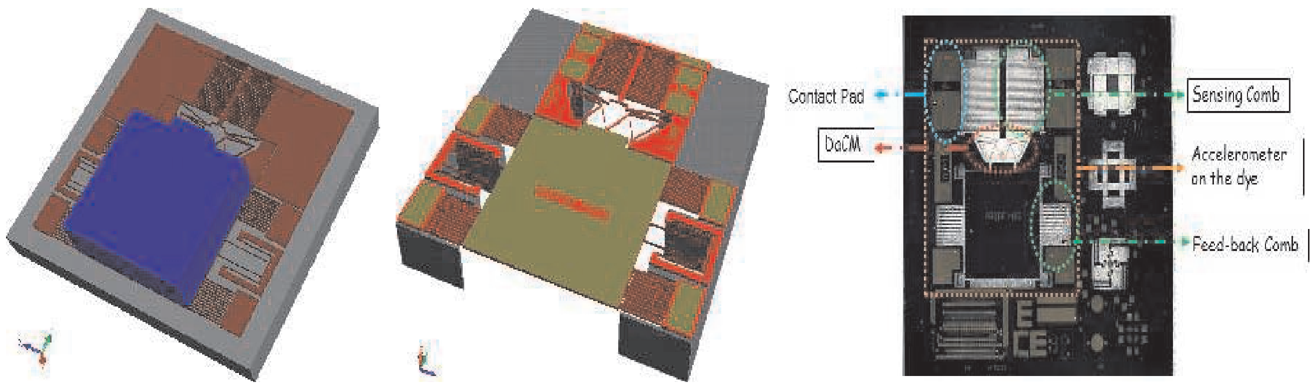


Figure 29: (a) The system-on-board packaged SOIMUMPs die, (b) The die on the PCB after wire-bonding.

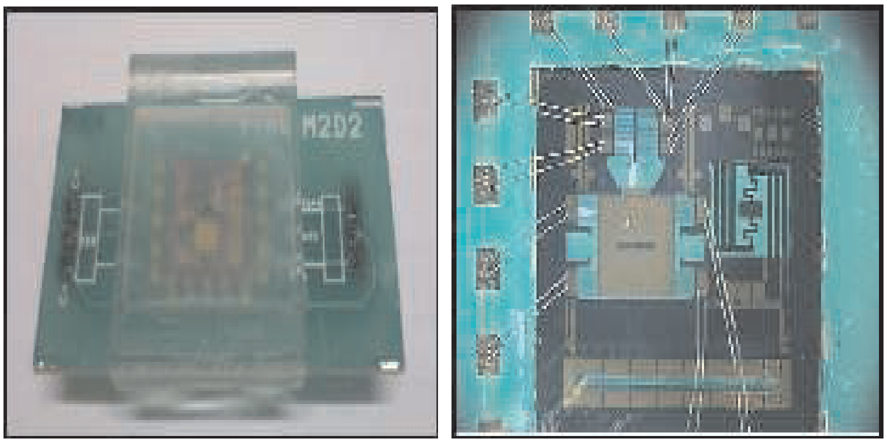
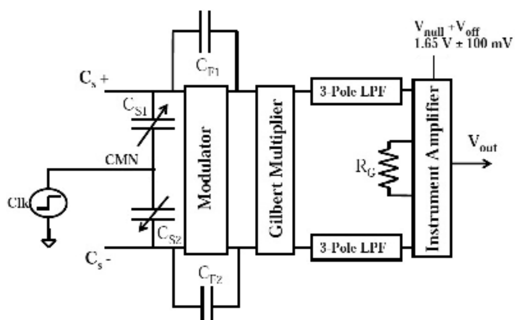


Figure 30: Block diagram of signal conditioning circuit prototyped on 0.35 μm technology.



$$V_{out} = 1.65 + (V_{carr} \cdot G \cdot (\Delta C / C_F) \cdot \cos(\omega_{signal}\, t))$$

contamination. The packaged die is shown in the Figure 29.

The packaged SOIMUMPs accelerometer with differential comb-drive senses external acceleration by changing the base capacitance of the comb-drive. A signal conditioning circuit called a Capacitance Sensor is required to convert the change in capacitance to change in voltage. Two different capacitance sensors were used for integration to the packaged SOIMUMPs accelerometer: (1) A MS3110 Universal Capacitive Readout IC [57], (2) A Universal Capacitance Sensor ASIC [32], and developed as part of this project. The inputs to these Capacitance Sensors are the C^+ and C^- ports of the differential comb-drive capacitance. The accelerometer integrated to the Capacitance sensing board was placed on a LDS shaker table which can shake at a given acceleration. A commercially available 3-axis analog milli-g accelerometer, made by ST Microelectronics (Model: LIS3L02AS4), was used as a standard accelerometer for calibrating the SOIMUMPs accelerometer and controlling the shaker table.

The capacitance to analog voltage conversion ASIC developed on 0.35 μm technology platform can sense single ended or differential capacitance change up to sub-femto range. The ASIC based on Continuous Time Voltage sensing [51] methodology is robust to the effect of parasitics largely present in the system and exhibits a static sensitivity of 30 mV/fF for a gain of 10. The signal conditioning has a modulation and a coherent demodulation module with programmable features such as the bandwidth and the sensor offset correction to customize the ASIC for a specific MEMS transducer. The signal conditioning block diagram is as shown in Figure 30. The packaged milli-g accelerometer was interfaced with the IISc ASIC and then tested on a shaker table. The mounting arrangement is shown in the left and centre panels of Figure 31, and the dynamic response of the accelerometer is shown in the right panel. Due to packaging, parasitics, and noise, the minimum acceleration measured was 30 mg [40]. Efforts are underway to improve the performance and also fabricate the device with a thick proof-mass to achieve micro-g resolution.

Noise figure: A measure of the amount of noise that the receiver adds to the radio signal coming from the antenna.

Signal to noise ratio (SNR): The amount of signal power per unit noise power, usually measured in decibels.

Bit error rate (BER): The fraction of bits, amongst all those transmitted, that are received in error.

9. Towards low power WSN motes

9.1. *A low power adaptive radio*

In a typical mote used in a wireless sensor network, the radio transceiver module is the dominant power consumer. For example, in Texas Instrument's chipset for a mote, the radio current is about 20 mA each for both receiver and transmitter, while that for the rest of the micro-controller functions is less than 5 mA [14,1]. We will focus on techniques we have developed to reduce the power of the receiver (see also Section 10.1). We will first give a brief overview of a typical receiver architecture and then introduce two main ideas we have explored to reduce the receiver power, followed by some simulation and measurement results.

The key parameters in the design of the receiver are the noise figure and linearity. The noise figure indicates how much noise the receiver adds to the RF signal obtained from the antenna. The requirement is that the added noise, and hence the consequent degradation in signal to noise ratio (SNR) be sufficiently small such that the target bit error rate (BER) is still achieved. The added noise in the analog section can be reduced by using larger currents to bias the circuits. In the digital section, by increasing bit-width and sampling rate, the added digital noise can be reduced, albeit at an increased power consumption. From the desired BER, one then derives the maximum noise figure and the minimum linearity and proceeds to design the receiver by burning the requisite amount of current to achieve this. However such a methodology leads to a very conservative design for the situations when the received signal strength is much larger or there is not much interference in the neighbouring frequency bands. A key contribution of our work is to develop a technique to make the receiver *power scalable,* that is, it reduces its power consumption if the received signal strength and interference are better than expected [21].

Figure 31: Left panel: DaCM accelerometer interfaced onto the ASIC; centre panel: Integrated system mounted on the shaker table to sense milli-g acceleration; right panel: Dynamic Response of the integrated system developed at IISc for an input acceleration of 400 mg at 20 Hz.

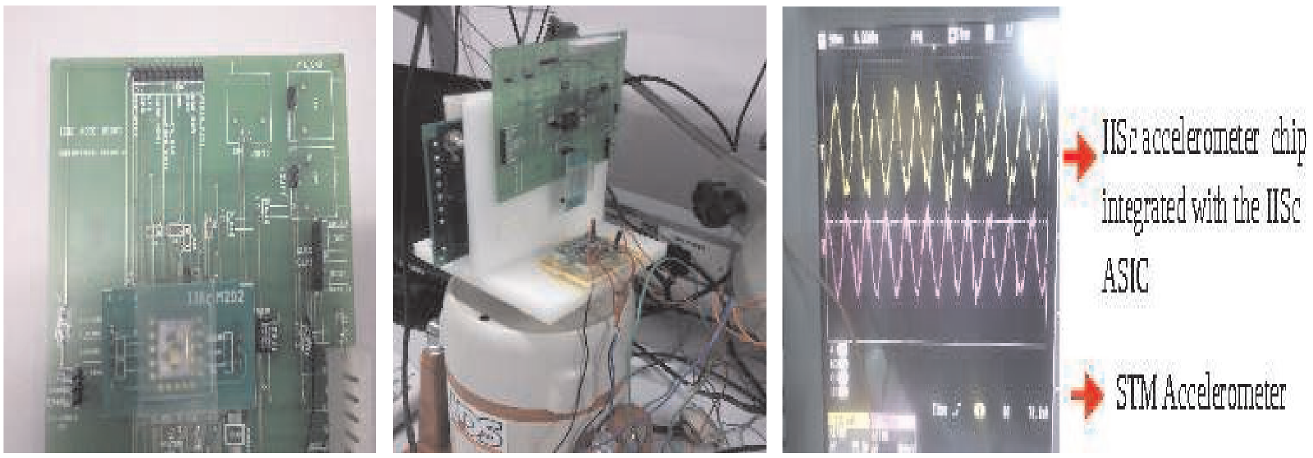
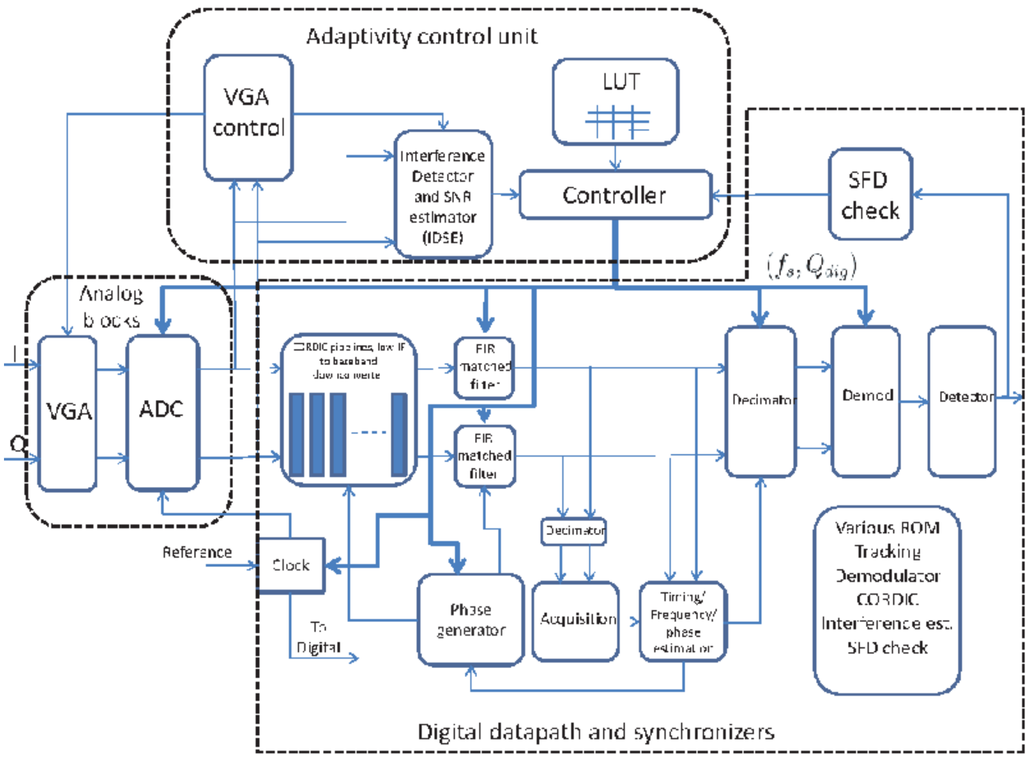


Figure 32: Power scalable digital backend design.


Adaptivity control unit
VGA control
LUT
Interference Detector and SNR estimator (IDSE)
Controller
SFD check
(f_s, Q_{dig})
Analog blocks
I
Q
VGA
ADC
CORDIC pipelines, low IF to baseband down convert
FIR matched filter
FIR matched filter
Decimator
Demod
Detector
Decimator
Various ROM Tracking Demodulator CORDIC Interference est. SFD check
Reference
Clock
To Digital
Phase generator
Acquisition
Timing/ Frequency/ phase estimator
Digital datapath and synchronizers

Sampling rate: The rate at which the received analog signal is sampled and discretised in time.

Digital bit width: The number of bits used to represent the sampled analog value.

9.1.1. *Power scalable receiver design*

A variable gain amplifier (VGA) is a commonly used block in any receiver analog front end. The variable gain amplifer adjusts the gain in the signal path such that a full scale voltage can be maintained at the inputs of the ADC. Thus, when the received signal strength is large, gain is reduced and vice versa. However, while this gain adjustment is fairly common, we propose to adjust the power consumption of the entire receiver in converse to the received signal strength. Furthermore, we also look at the strength of the interfering signals in the neighbouring band to determine how to adjust the overall power consumption. Figure 32 shows our power scalable receiver design for the digital back end. The main additional components for making the receiver power scalable are: a) tuning knobs for the ADC and the digital datapath to dial down the power consumption as required b) an adaptivity control unit consisting of a signal and interference estimation module and a scalable filter design.

Scalable power control for the ADC and the digital data path

The two main knobs for trading off power with performance are the sampling rate and the digital bit width. The sampling rate is adjusted by adjusting the frequency of the clock being used by these sections. The maximum rate is 30 Msps and the minimum is 2 Msps. The higher sampling rate allows for better averaging of the noise and hence results in better BER, but at the cost of increased dynamic power dissipation. The bit resolution is adjusted by selectively masking the bits at the input to the digital data path (i.e. the bits from the ADC are ANDed and shifted down). For example, the maximum bit resolution is 8-bits and the entire data path is designed for this. However, under certain benign conditions, even a 1-bit data path suffices. To accommodate this, the lower 7-bits from the ADC are masked and the 8th bit is right shifted all the way to the first bit and fed into the data path. The same is done for all the other input signals in the datapath — like the input from the digital oscillator. This ensures that the switching activity is restricted to only the lower order bits, while the higher order bits remain inactive, thus saving dynamic power.

Adaptivity control unit

The adaptivity control unit measures the interference and signal power levels during the preamble of the packet in a non-coherent manner. The measurement estimates are then used to determine the operating parameters for the receiver for receiving the data symbols of the packet following the preamble.

The two key components of the adaptivity control unit are the interference and signal power estimation module and the lookup table which determines the receiver's operating parameters. The 802.15.4. standard specifies the BER requirements for the case when the signal amplitude is the weakest at – 85 dBm, the adjacent channel has a power of – 85 dBm and the alternate channel has a power of – 55 dBm. The baseline sampling rate of 30 Msps and resolution of 8-bits for the ADC and digital data path ensures that the target BER is met under these conditions. When there is not much interference or the received signal is stronger, we detect this condition and dial down the sampling rate and resolution down up to 2 Msps and 1-bit. We determine the energy of the adjacent and alternate bands by downconverting them and measuring the power over a duration of 4 chips. Each chip is a half-sine waveform modulated as ± 1. Sixteen chips each for the in-phase (I) and quadrature (Q) components forms one 4-bit symbol in the 802.15.4 standard. Signal power is also similarly measured. The same hardware is reused for all these measurements to save on gate count. The measured interference power is quantized to 1-bit each for adjacent and alternate channels and a 6-bit quantization is used for representing the signal power. These 8-bits are then used to look up into a precharacterized table, which tells the right sampling rate and quantization to use for receiving the signal under these conditions, while meeting BER. This lookup table is characterized a priori using MATLAB.

dBm: The decibel value of the measured power in milliwatts.

Scalable filter design

When the sampling rate is adjusted to be less than its maximum values, the filter in the data path gets affected and special attention needs to be paid to ensure that the overall transfer function is still maintained, invariant to the sampling/operating rate. The filter used is a matched filter, and is implemented by correlating the received sequence with the samples of the half-sine modulating pulse for the chip. This pulse was generated by a CORDIC [4] algorithm. When the sampling rate is 30 Msps, all the filter taps are used. Since the filter coefficients are symmetric about the central value, the number of multipliers is reduced to half. When the sampling rate is reduced from 30 Msps to say 3 Msps, then only three taps are kept active, with zeros fed as inputs to the other taps. This scales the dynamic power dissipation accordingly.

Table 3: Mica2 platform current draw, measured with a 3V power supply.

Device/Mode	Current (mA)	Device/Mode	Current (mA)
CPU		*Radio*	
Active	8.0	Receiving	7
Idle	3.2	Transmitting(-20 dBm)	3.7
ADC Acquire	1.0	Tx(-8 dBm)	6.5
Extended Standby	0.223	Tx(0 dBm)	8.5
Standby	0.216	Tx(10 dBm)	21.5
Power-save	0.110	*Sensors*	
Power-down	0.103	Typical board	0.7

9.1.2. *Current status*

The design of the receiver has been implemented in the Verilog Hardware Description Language (HDL) and has been synthesized for the UMC 130 nm process technology. The power for the different operating conditions was estimated using Synopsys Power Compiler. Savings of 85% in power were obtained when the receiver is operated at its minimal configuration (for benign condition) as compared to the worst case setting for the harshest input conditions.

Our main motivation is to take advantage of benign channel conditions and dial down the power consumed by the receiver. We achieve this by incorporating an interference and signal estimator block which measures the signal and interference levels during the preamble. A precharacterized lookup table is then consulted to set the sampling rate and bit resolution of the digital data path to be as low as possible for the rest of the packet reception. We also implement the timing recovery algorithms in a streaming fashion, to minimize the power dissipation associated with storage. With these techniques, a receiver in a UMC 130nm process shows upto 85% power reduction between its maximum and minimum power settings, thus showing power scalability.

10. Mote system software

10.1. *Compiler assisted energy management for sensor network nodes*

The major sources of power consumption in a sensor network node are the transmission unit and the computing unit. Table 3 adapted from the study done in [71] shows the typical current drawn in various units in a sensor network node. From Table 3 we observe that indeed the power consumed by a radio is significant, and power required for computation, though lower, is quite significant. Given the ability to operate the processor in both active and idle mode these numbers do not present the complete picture. The computational demands of the application determine the CPU's actual

Dynamic voltage scaling (DVS): A power management technique that increases or decreases the voltage of a component system as per the need for use of the component.

Cache: A component that stores data so that future requests for the same data can be served faster.

Scratchpad: A high-speed internal memory for temporarily storing intermediate calculations.

Embedded system: A computing system, embedded in a larger system, and designed for specialised functions with real-time computing constraints.

activity and radio usage. In [71], the authors also present a detailed breakdown of energy consumed by different components for a variety of applications. The results suggest that CPU power ranges from 28% to 86% of the total power consumed and roughly 50% on average. Applications like data filtering, encryption and decryption (required in networks used for critical tasks such as military surveillance) and more efficient communication protocols can shift the activity from the radio to the CPU. Hence, saving CPU energy will also play a crucial role.

Researchers in the sensor network domain have focused more on optimizing energy in the communication system than in the computing system. Researchers have tackled the problem of energy optimization in communication system, from various aspects. Some proposals try to make use of power aware radios [73,76], while other use more efficient modulation schemes [69] and energy aware sensor network protocols [33,63,78]. Researchers have also suggested using energy aware software (see Section 9). For example, implementing dynamic power management through operating system [10,35]. Recently, there has been efforts for minimizing energy consumption in computing systems as well. In this respect researchers have proposed low power architecture design for sensor network node processors [46,22,34,59]. These designs suggest use of specialized hardware units for performing tasks in a sensor network node.

An important requirement of the energy management techniques used for sensor networks is that the extra computation and communication introduced by the energy optimization schemes must be low. Otherwise, the energy required to perform the optimization schemes may outweigh the benefits. Conventional techniques in sensor network domain try to minimize communication energy, as conventional communication links consume a significant amount of energy. However, recent developments in self-powered MEMS-based RF communication devices [49] can lead to sensor networks where the communication link is entirely self-powered. Hence, computational energy will become even more important.

We have focused on compiler assisted energy optimization techniques for sensor network node processors. To the best of our knowledge little work has been done in compiler related optimization techniques for sensor networks [46]. In compiler assisted energy optimization strategies major decisions are taken offline. During runtime, based on the current system state, the appropriate decision is invoked. On the other hand, for OS level techniques, all the decisions are to be made at runtime, incurring energy cost on already energy constrained nodes. Our main contribution are outlined as follows.

Compiler assisted dynamic voltage scaling

Different sensing applications will have different processing requirements in the active state. Having a processor with a fixed throughput is necessarily power inefficient. Having a custom digital signal processor for every sensing application is not feasible both in terms of the cost and time overhead. However, energy savings can still be obtained by the use of dynamic voltage scaling. We have implemented and evaluated a compiler directed dynamic voltage scheme for sensor network nodes. The implemented technique leads to an average energy savings of 29.04% at a performance slow down of 2.88%. On average 26.99% of improvements in energy-delay product are observed.

Compiler assisted scratchpad memory allocation

Memories are a major consumer of energy in a system. They also limit the performance of the processors. In general purpose processors caches have been used to reduce the memory access time. Caches are power hungry and also introduce non-determinism in the system. Hence, embedded systems use more energy efficient scratchpad memories. In this project we show the benefits of using scratchpad memory in sensor network node processors. We have implemented a compiler assisted scratchpad allocation algorithm. On an average 50% of energy savings and 52% performance improvements are obtained with a 512 byte of scratchpad memory. We also found that the scratchpad size for a system should be chosen carefully, as it may have a negative impact on energy savings.

We also studied the effect of the presence of scratchpad memory on the dynamic voltage scaling. Our results show that the presence of scratchpad memory in the system limits the opportunities for dynamic voltage scaling. We also performed a comparison among all the three optimization cases (DVS, Scratchpad, Scratchpad and DVS together) normalized to the no-optimization case. Our comparison results show that benefits of a scratchpad memory in a system are comparable to the benefits of scratchpad and DVS applied together and improve further with an increase in scratchpad size.

10.2. *Thread-based sensor node OS with limited stack*

Sensor nodes exhibit characteristics of both embedded systems and general-purpose systems. A sensor node operating system is similar to an embedded operating system, but unlike in a typical embedded operating system, sensor network applications may not be real time, and are constrained by limited memory, computational power and energy available in each sensor node. The computational power and program memory is restricted to minimize power consumption as the motes run on battery power. Therefore, it is important to design an operating system for sensor nodes that attempt to minimize the use of memory allocated at runtime, i.e., dynamic memory such as stack. Other basic requirements of wireless sensor network OS are the following.

Multithreading: A programming paradigm that allows multiple threads of computation to coexist and share the resources assigned to a process.

Reliable computation and communication

In order to ensure reliable computation, it is necessary to detect faults such as stack overflow. If dynamic memory is not used for stack, it may possibly be easier to detect stack overflow as the position of stacks would be fixed. It is also important to ensure end-to-end reliable communication for the proper functioning of the WSNs.

Preemptive scheduling: A scheduling paradigm in which a task may be interrupted in order to serve another task of higher priority.

User-friendly programming environment

There are two aspects of user-friendly programming environment, viz., (i) programming using well known programming language, such as C, and using well known programming paradigm like thread paradigm, and (ii) a good programming interface in the programming paradigm used. Most of the OSs for WSNs [68] have been written in C, but they may not use thread paradigm because of the use of a stack for every thread. A few other programming languages such as nesC [18], galsC [26], etc., have also been proposed for writing OSs for WSNs. In TinyOS [48], both the OS and the applications have been written in nesC which is harder to learn, and therefore adds to the additional problem of developing and debugging the system.

Stack: A segment of memory associated with a process, and with last-in-first-out access.

Preemptive multiprogramming

Even though the applications for WSNs may not be real time, some of the distributed actions of the OS for WSNs may have to be synchronized, and therefore, it is desirable to support preemptive multiprogramming along with thread programming paradigm. It is also necessary that the thread programming paradigm has a good programming interface.

By considering the above requirements, we have proposed a thread-based execution model that uses only a fixed number of stacks. In this execution model, the number of stacks at each priority level is fixed. It minimizes the stack requirement for multi-threading environment and at the same time provides ease of programming. We also take advantage of the fixed position of stacks to detect stack overflow, and therefore increase the reliability of the OS for sensor nodes. We give a separate implementation of this thread model in the Contiki OS [19] without using any part of protothread implementation in the Contiki OS.

10.2.1. *Proposed model and implementation*

Due to the presence of long running tasks and some basic real time requirements of sensor node OS, such an OS should allow preemption and priority-based scheduling. We have chosen Contiki as our development platform, as it provides many features like dynamic loading and unloading of programs and services. Keeping the basic event driven model of Contiki kernel as it is, we embedded within it our thread-based model. In our model, the number of priority levels (MAX_LEVELS) (i.e., priority queues) is fixed. Depending upon the memory available, we have also fixed the number of stacks for each priority level (STACKS_PER_LEVEL), and from this pool of stacks, a stack is allocated to a thread in a priority level before it gets scheduled. Therefore, at any point of time, the number of threads in Ready state is upper bounded by the maximum number of stacks.

Total stack space required by the system is bounded by treating it as a limited resource. Stack must be allocated to a thread before moving it to LT_READY state as shown in Figure 33. The stack will be released explicitly either by the thread blocking for an event or by the system when the thread exits. It is required that the thread in a priority level either always hold the stack or always release the stack, while waiting for an event. If a priority level requires the threads to hold the stack while waiting for an event, then the priority is said to be a stack holding priority; otherwise, the priority is said to be a stack releasing priority.

Initially, when a thread is created (LT_CREATED state), the stack is not allocated to it. It will be moved from LT_CREATED state to LT_READY state only when stack is allocated for this thread; otherwise it is moved to LT_WAITING_STACK state. If a thread is in LT_RUNNING state with stack holding priority, it can wait for an event holding the stack. Similarly, if a thread is in LT_RUNNING state with stack releasing priority, it can wait for an event releasing the stack. If the thread is in LT_BLOCKED_RELEASE state, and the event occurs, the thread is moved from LT_BLOCKED_RELEASE state to LT_WAITING_STACK state or to LT_READY state depending upon the availability of the stack. Another possibility is that when a thread wants to change its priority either to stack holding priority or to stack releasing priority, it directly moves from LT_RUNNING state to LT_WAITING_STACK state. When a thread releases stack, and later on it obtains a stack again, the location of the new stack may be different from that of the previous stack allocated to it as we can have multiple stacks at each priority level.

A typical application of release stack and wait for an event can be a thread that waits for encrypting

Figure 33: The new state transition diagram.

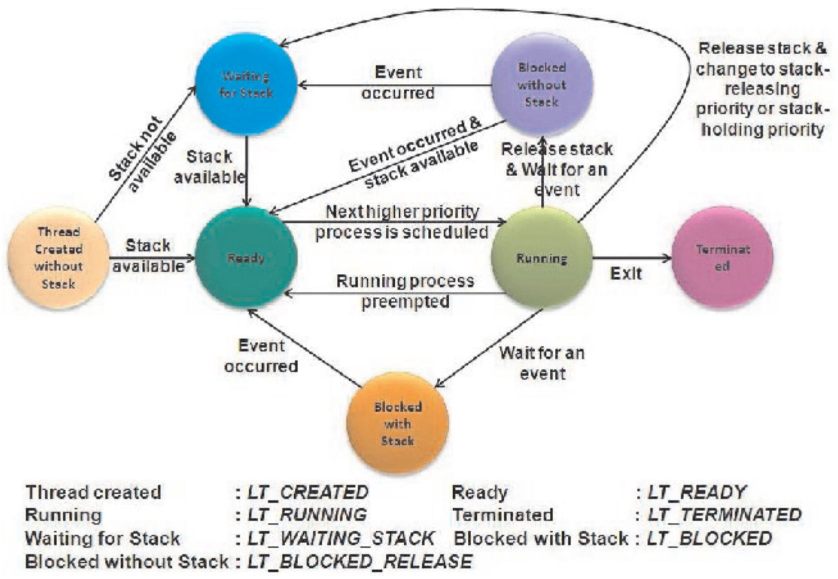
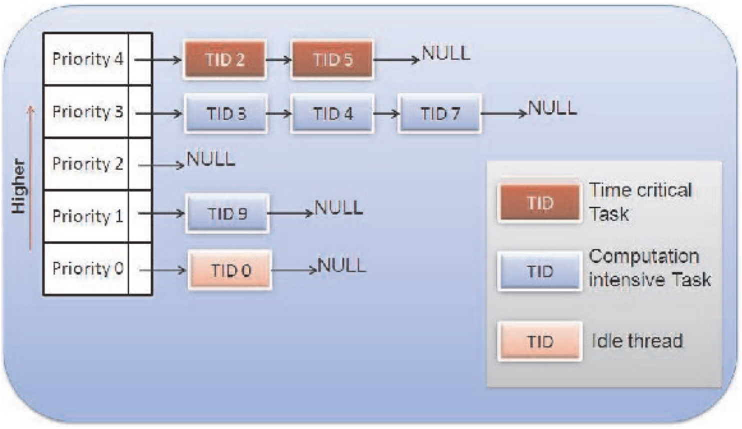

Waiting for Stack
Blocked without Stack
Event occurred
Release stack & change to stack-releasing priority or stack-holding priority
Stack not available
Stack available
Event occurred & stack available
Release stack & Wait for an event
Thread Created without Stack
Stack available
Ready
Next higher priority process is scheduled
Running
Exit
Terminated
Running process preempted
Event occurred
Wait for an event
Blocked with Stack
Thread created : LT_CREATED
Ready : LT_READY
Running : LT_RUNNING
Terminated : LT_TERMINATED
Waiting for Stack : LT_WAITING_STACK
Blocked with Stack : LT_BLOCKED
Blocked without Stack : LT_BLOCKED_RELEASE

Figure 34: Snapshot of threads active in system for one stack per level.



a message. Once the encryption of the message is done, it can choose to release the stack, and wait for the next message to be encrypted. The advantage of having the option of releasing stack and waiting is that we do not need to waste resources on the overhead of creating context of threads again and again for repetitive tasks.

10.2.2. *Scheduling policy*

Our proposed model uses the following scheduling policy.

- Fixed priority preemptive scheduling policy is used.
- FIFO is used for threads having the same priority.
- If a thread with stack releasing priority moves to a stack holding priority, it does not hold any resources.
- If a thread with stack holding priority moves to a stack releasing priority, it first releases the stack, but can continue to hold other resources.

The implication of the above scheduling policy is the following.

- A low priority thread is scheduled only if no higher priority thread is present in ready state.
- The currently running thread is preempted if another thread with priority strictly greater than that of the currently executing thread is moved to ready state.

The number of priority levels as shown in Figure 34 are fixed but can be changed at compile-time. Priority levels cannot be added or removed at runtime to reduce overhead of cleaning priority queues and managing priority conflict. Using FIFO scheduling policy among the threads with the same priority, the number of preemption needed will be much lesser compared to that needed by a round robin scheduling policy. As the number of context switches decreases, the energy consumption will also decrease. Using this scheduling policy, we can take maximum advantage of dynamic voltage scaling, as the threads in a priority level with non-real time requirements can be executed at a slower pace.

There is no possibility of deadlock due to contention for stack in the above scheduling scheme. This is because a thread in stack holding priority level does not hold a resource while waiting for a stack. Therefore, no other threads will wait for it, while it waits for stack. A thread in stack releasing priority level can hold resources while waiting for stack, but it is always assured of getting a stack as other threads in the same priority level with ready or running state will continue to be executed until they release their stack.

Thread 0 is the default thread for system with lowest priority, and it monitors the system. If there are no other threads to be scheduled, it switches to power saving mode if supported by the microprocessor.

Our thread model has been implemented without using any part of protothread implementation, and applications using our thread model do not need to use protothreads. However, protothreads can be used together with our threads as higher priority threads using the stack space of the event driven kernel stack.

11. Conclusion

We have reported on the outcomes of a research and demonstration project on human intrusion detection in a large secure space using an ad hoc wireless sensor network. As one would have expected, there were aspects of our work specific to this application, and other general aspects relevant to a wider set of applications. After comparing acoustic, vibration, and passive infrared (PIR)) sensors, we found PIR sensors to be the most effective for our application. A PIR "trip-wire" was developed by wirelessly networking several mote "platforms", each with three off-the-shelf quad-pixel PIR devices. Each such platform is able to cover a 360° field of vision, and is equipped with a support vector machine derived classification algorithm, which is trained to distinguish human intrusion from vegetation "clutter". Several such parallel trip-wires help reduce the probability of false alarm. Such a detection system is connected to the operator station by a multihop wireless ad hoc network of low power relay nodes. The network was designed to be self-organising, self-routing, and self-healing. Several security features have also been included in our design. In addition to the demonstrable system, which we call SmartDetect, the project has yielded new basic research in the areas of self-healing geographical routing, distributed sequential algorithms for change detection in random processes, a MEMS accelerometer with a capacitive output, and a femto-Farad capacitance measurement circuit, that can be used for footstep detection, the design of and some components of a micro-radio that can adjust its power consumption depending on the signal-to-noise ratio of the signal it receives, and energy efficient embedded systems software.

Considerable engineering work remains to be done to convert SmartDetect into a robust deployable system. We propose to follow up on some of the scientifically interesting challenges in a possible second phase of the project. One of the areas where wireless sensor networking can be expected to be of significant commercial importance is in industrial sensor networking, where 100s of sensors in an industrial setting need to be interconnected wirelessly. This problem throws up the challenge of designing low cost and low power wireless mesh networks with predictable performance in terms of delay and reliability. This is another challenge that we are currently pursuing in our work.

Acknowledgement

We would like to thank Prof. V.K. Aatre (visiting professor ECE, IISc, and former Scientific Advisor to the Defence Minister) for his support and encouragement, and Dr. S.V. Sankaran, Director ER&IPR, DRDO, for his support in the many administrative matters that we encountered during execution of the project. The DRDO lab we interacted with in the course of this project was the Centre for Artificial Intelligence and Robotics (CAIR). We would like to thank Dr. N. Sitaram, former Director of CAIR, and Dr. V.S. Mahalingam, current Director of CAIR. Our principal point of contact at CAIR was Dr. N. Ramamurthy. We thank him and his team of engineers for meeting us frequently, and providing very constructive feedback. Prof. A. Arora from Ohio State University, was a visiting faculty member to our project, and provided a wealth of useful suggestions. Thanks are also due to Dr. A. V. Krishna from PESIT, Bangalore, for introducing Support Vector Machines to this team.

Received 14 May 2010; revised 15 October 2010.

References

1. IEEE 802.15.4. Part 15.4 : Wireless medium access control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area networks (LR-WPANs). October 2003.
2. P. Agrawal and J. Kuri. A Key Distribution Scheme without Deployment Knowledge. In *Proceedings of the International Conference on Ultra-Modern Telecommunications (ICUMT)*, 2009.
3. J. Altman. Acoustic and seismic signals of heavy military vehicles for co-operative verification. *Journal of Sound and Vibration*, 273:713–740, 2004.
4. Ray Andraka. A survey of CORDIC algorithms for FPGA based computers. In *FPGA '98: Proceedings of the 1998 ACM/SIGDA sixth international symposium on Field programmable gate arrays*, pages 191–200, New York, NY, USA, 1998. ACM Press.
5. Applied MEMS, Inc., www.appliedmems.com.
6. Anish Arora, P. Dutta, S. Bapat, V. Kulathumani, H. Zhang an V. Naik, V. Mittal, H. Cao, M. Demirbas, M. Gouda, Y. Choi, T. Herman, S. Kulkarni, U. Arumugam, M. Nesterenko, A. Vora, and M. Miyashita. A line in the sand: a wireless sensor network for target detection, classification and tracking. *Journal of Computer Networks*, 2004.
7. Anish Arora, Emre Ertin, Prasun Sinha, Ted Herman, Nishank Trivedi, Mikhail Nesterenko, Mohamed Gouda, David Culler, Mike Grimmer, and Ken Parker. Exscal: Elements of an Extreme Scale Wireless Sensor Network. Technical report, The Ohio State University, University of Iowa, Kent State University, Michigan State University, The University of Texas at Austin, and University of California at Berkeley, 2004.
8. Anish Arora, Rajiv Ramnath, Emre Ertin, Prasun Sinha, Sandip Bapat, Vinayak Naik, Vinod Kulathumani, Hongwei Zhang, Hui Cao, Mukundan Sridharan, Santosh Kumar, Nick Seddon, Chris Anderson, Ted Herman, Nishank Trivedi, Chen Zhang, Mikhail Nesterenko, Romil Shah, Sandeep Kulkarni, Mahesh Aramugam, Limin Wang, Mohamed Gouda, Young ri Choi, David Culler, Prabal Dutta, Cory Sharp, Gilman Tolle, Mike Grimmer, Bill Ferriera, and Ken Parker. Exscal: Elements of an extreme scale wireless sensor network. In *Proceedings of the 11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications(RTCSA)*, 2005.
9. Taposh Banerjee, V. Kavitha, and V. Sharma. Energy efficient change detection over a MAC using physical layer fusion,. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Las Vegas, USA, 2008. IEEE.
10. Luca Benini, Giuliano Castelli, Alberto Macii, and Riccardo Scarsi. Battery-driven dynamic power management. *IEEE Des. Test*, 18(2):53–60, 2001.
11. A. B. Bhaskar, Girish Krishnan, N. Shamsudhin, and G.K. Ananthasuresh. Design, fabrication, and tetsing of a meso-scale accelerometer made of spring steel. *Journal of the Instrumentation Society of India*, 39(1):46–52, March 2009.
12. Abhijit Bhattacharya and Anurag Kumar. Delay constrained optimal relay placement for planned wireless sensor networks. In *Proc. IWQoS 2010: The 18th IEEE International Workshop on Quality of Service*. IEEE, June 2010.
13. Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A training algorithm for optimal margin classifiers. In *Proc. of the Fifth Annual Workshop on Computational Learning Theory*, pages 144–152. ACM Press, 1992.
14. B. Bougard, F. Catthoor, D. C. Daly, A. Chandrakasan, and W. Dehaene. Energy efficiency of the ieee 802.15.4 standard in dense wireless microsensor networks: Modeling and improvement perspectives. In *Design Automation and Test in Eurpoe (DATE)*, March 2005.
15. Michael Buettner, Gary V. Yee, Eric Anderson, and Richard Han. X-mac: a short preamble mac protocol for duty-cycled wireless sensor networks. In *SenSys '06: Proceedings of the 4th international conference on Embedded networked sensor systems*, pages 307–320, New York, NY, USA, 2006. ACM.
16. Costas Busch, Srikanth Surapaneni, and Srikanta Tirthapura. Analysis of link reversal routing algorithms for mobile ad hoc networks. In *SPAA '03: Proceedings of the fifteenth annual ACM symposium on Parallel algorithms and architectures*, pages 210–219, New York, NY, USA, 2003. ACM.
17. Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. Software available at http://www.csie.ntu.edu.tw/∼cjlin/libsvm.
18. Elaine Cheong and Jie Liu. galsc: A language for event-driven embedded systems. In *DATE '05: Proceedings of the conference on Design, Automation and Test in Europe*, pages 1050–1055, Washington, DC, USA, 2005. IEEE Computer Society.
19. A. Dunkels, B. Gronvall, and T. Voigt. Contiki — a lightweight and flexible operating system for tiny networked sensors. In *Local Computer Networks, 2004. 29th Annual IEEE International Conference on*, pages 455–462, Nov. 2004.
20. A. Duwel and Barbour. Mems development at draper laboratory. In *Proceedings of the SEM Annual Conference, Charlotte, NC*, June 2–4 2003.
21. Satyam Dwivedi. *Low Power Receiver Architecture and Algorrithms for Low Rate Wireless Personal Area Network*. PhD Thesis, IISc, 2010.
22. Virantha Ekanayake, IV Clinton Kelly, and Rajit Manohar. An Ultra Low-Power Processor for Sensor Networks. *SIGARCH Comput. Archit. News*, 32(5):27–36, 2004.
23. A. Ekimov and Sabatier, J.M. Ultrasonic wave generation due to human footsteps on the ground. *Journal of the Acoustical Society of America*, 21(3):EL114–119, March 2007.
24. N. Furestenau, M. Schmidt, H. Horack, W. Goetze, and W. Schmidt. Extrinsic fabry-perot interferometer vibration acoustic senso systems for airport ground traffic monitoring. In *Proc. IEE on Optoelectronics*, volume 144, pages 134–144, 1997.
25. Eli M. Gafni, Student Member, Dimitri, P. Bertsekas, and Senior Member. Distributed algorithms for generating loop-free routes in networks with frequently changing topology. *IEEE Transactions on Communications*, 29:11–18, 1981.
26. David Gay, Philip Levis, Robert von Behren, Matt Welsh, Eric Brewer, and David Culler. The nesc language: A holistic approach to networked embedded systems. In *PLDI '03: Proceedings of the ACM SIGPLAN 2003 conference on Programming language design and implementation*, pages 1–11, New York, NY, USA, 2003. ACM.
27. Lin Gu, Dong Jia, Pascal Vicaire, Ting Yan, Liqian Luo, Ajay Tirumala, Qing Cao, Tian He, John A. Stankovic, Tarek Abdelzaher, and Bruce H. Krogh. Lightweight detection and classification for wireless sensor networks in realistic environments. In *Proceedings of the 3rd international conference on Embedded networked sensor systems (SenSys)*, Nov. 2005.
28. A. Gupta and J. Kuri. Deterministic Schemes for Key Distribution in Wireless Sensor Networks. In *Proceedings of the Third International Conference on COMmunication System softWAre and middlewaRE (COMSWARE*, Bangalore, India, January 2008.
29. A. Gupta, J. Kuri, and P. Nuggehalli. A New Scheme for

Establishing Pairwise Keys in a Wireless Sensor Network. In *Lecture Notes in Computer Science (LNCS), International Conference on Distributed Computing and Networking (ICDCN)*, December 2006.

30. A. Gupta, P. Nuggehalli, and J. Kuri. An Efficient Scheme for Establishing Pairwise Keys in a Wireless Sensor Network. In *Proceedings of the First Workshop on Wireless Systems: Advanced Research and Development (WISARD)*, Bangalore, India, January 2007.
31. Tian He, Sudha Krishnamurthy, Liqian Luo, Ting Yan, Lin Gu, Radu Stoleru, Gang Zhou, Qing Cao, Pascal Vicaire, John A. Stankovic, and Tarek F. Abdelzaher. Vigilnet: An Integrated Sensor Network System for Energy-Efficient Surveillance. *ACM Mobisys*, 2004.
32. S. Hegde, Thejas, and N. Bhat. Universal capacitance sensor. In *International conference on smart structures and Systems*, pages 59–66, June 2008.
33. Wendi Rabiner Heinzelman, Anantha Chandrakasan, and Hari Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *Proceedings of 33rd Annual Hawaii International conference on System Scineces*, 2000.
34. Mark Hempstead, Nikhil Tripathi, Patrick Mauro, Gu-Yeon Wei, and David Brooks. An Ultra Low Power System Architecture for Sensor Network Applications. In *Proceedings of the 32nd annual international symposium on Computer Architecture*, pages 208–219. IEEE Computer Society, 2005.
35. Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David Culler, and Kristofer Pister. System architecture directions for networked sensors. *SIGPLAN Not.*, pages 93–104, 2000.
36. Hougen, D. F. et al. A miniature robotic system for reconnaissance and surveillance. In *Proc. 2000 IEEE International Conference on Robotics and AutomationSan, Francisco*, pages 501–507, April 2000.
37. Arunkumar Jayaprakasam and V. Sharma. Cooperative robust sequential detection algorithms for spectrum sensing in cognitive radio. In *Proc. International Conference on Ultra Modern Telecommunication (ICUMT)*, St. Petersburg, Russia, October 2009.
38. H. M. Yuichi Kaji and R. Matsumoto. Key Predistribution Schemes for Wireless Sensor Networks using Lines and Points over a Finite Geometry. In *Proceedings of IEEE SECON*, September 2006.
39. Brad Karp and H. T. Kung. Gpsr: greedy perimeter stateless routing for wireless networks. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 243–254, New York, NY, USA, 2000. ACM Press.
40. S. Khan, Thejas, N. Bhat, and G.K. Ananthasuresh. Design and characterization of a micromachined accelerometer with mechanical amplifier for intrusion detection. In *3rd ISSS National Conference on Smart and Micro Systems*, Kolkata, November 2009.
41. Shinji Kirihata, Katsuhiro Uchisawa, and Masao Yamaguchi. Low-profile dome-shaped multilens system. U.S. Patent 6,051,836, Apr. 18 2000.
42. G. Krishnan. Displacement-amplifying compliant mechanisms for sensor applications.
43. G. Krishnan and G. K. Ananthasuresh. A systematic method for the objective evaluation and selection of compliant displacement amplifying mechanisms for sensor applications. *Journal of Mechanical Design*, 130(Issue 10):102304:1–9, 2008.
44. G. Krishnan, C.U. Kshirasagar, and G.K. Ananthasuresh. Micromachined high-resolution accelerometers. *Journal of the Indian Institute of Science: a Multidisciplinary Reviews Journal*, 87(3):333–361, 2007.
45. Walter Kuster and Hans J. Keller. Domed segmented lens system. U.S. Patent 4,930,864, Jun. 5 1990.
46. Zhao Lei, Zhang W.-H., Xu C.-N., Xu Y.-J., and Li X.-W. Energy-Aware System Design for Wireless Sensor Network. *ACTA AUTOMATICA SINICA*, 32:892–899, 2006.
47. Ben Leong, Barbara Liskov, and Robert Morris. Geographic routing without planarization. In *NSDI'06: Proceedings of the 3rd conference on Networked Systems Design & Implementation*, pages 25–25, Berkeley, CA, USA, 2006. USENIX Association.
48. Philip Levis, David Gay, Vlado Handziski, Jan-Heinrich Hauer, Ben Greenstein, Martin Turon, Jonathan Hui, Kevin Klues, Cory Sharp, Robert Szewczyk, Joseph Polastre, Philip Buonadonna, Lama Nachman, Gilman Tolle, David Culler, and Adam Wolisz. T2: A second generation os for embedded sensor networks. Technical report, Telecommunication Networks Group, Technische Universität Berlin, November 2005.
49. H. Li and A. Lal. Radioisotape-Powered Cantilever for Vacuum Sensing with RF Transmission. In *Proceedings of 12th International Conference on Transducers, Solid-State Sensors, Actuators and Microsystems*, 2003.
50. H. Ling and T. Znati. End-to-end Pairwise Key Establishment using Multi-path in Wireless Sensor Networks. In *Proceedings of Globecom*, December 2005.
51. J.C. Lotters, W. Olthuis, P.H. Veltink, and P. Bergveld. A sensitive differential capacitance to voltage converter for sensor applications. *IEEE Transactions on Instrumentation and Measurement*, 48(1), February 1999.
52. Gopal Pandurangan Maleq Khan and V.S. Anil Kumar. Distributed algorithms for constructing approximate minimum spanning trees in wireless sensor networks. In *Parallel and Distributed Systems, IEEE Transactions*, volume 20, January 2009.
53. G.P. Mazarakis and J.N Avaritsiotis. A prototype sensor node for footstep detection. In *Proceedings of the Second European Workshop on Wireless Sensor Networks*. DOI: 10.1109/EWSN.2005.1462036, January 2005.
54. Yajun Mei. Information bounds and quickest change detection in decentralized decision systems. *IEEE Transactions on Information theory*, 51(7):2669–2681, July 2005.
55. R. N. Miles, R. Robert, and R. R. Hoy. Mechanically coupled ears for directional hearing in parasitoid fly ornia ochracea. *Journal of the Acoustical Society of America*, 98(6):3059–3070, 1995.
56. K. Miller, A. Cowen, G. Hames, and B. Hardy. Soimumps design handbook, memscap, revision 4.0., 2004.
57. MS3110 Data sheets. http://www.irvine-sensors.com/pdf/ms3110%20datasheet%20USE.pdf.
58. Raghuraman Mudumbai, Gwen Barriac, and Upamanyu Madhow. On the feasibility of distributed beam forming in wireless networks. *IEEE Transactions on Wireless Communications*, 6(5):1754–1763, May 2007.
59. Leyla Nazhandali, Bo Zhai, Javin Olson, Anna Reeves, Michael Minuth, Ryan Helfand, Sanjay Pant, Todd Austin, and David Blaauw. Energy Optimization of Subthreshold-Voltage Sensor Network Processors. *SIGARCH Comput. Archit. News*, 33(2):197–207, 2005.
60. I. V. Nikiforov. A generalized change detection problem. *IEEE Transactions on Information theory*, 41(1):171–187, Jan 1995.
61. E. S. Page. Continuous inspection schemes. *Biometrika*, 41(1/2):100–115, June 1954.
62. Panasonic Electric Works Corporation of America, New Jersey, USA. *MP Motion Sensor (AMN 1,2,4) Data Sheet.*
63. Joseph Polastre, Jason Hill, and David Culler. Versatile low power media access for wireless sensor networks. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 95–107. ACM, 2004.
64. K. Premkumar and Anurag Kumar. Optimal sleep-wake scheduling for quickest intrusion detection using sensor networks. In *Proc. IEEE Infocom*, Arizona, USA, Apr 2008.
65. K. Premkumar, Anurag Kumar, and Joy Kuri. Distributed detection and localization of events in large ad hoc wireless sensor networks. In *Proc. Forty-Seventh Annual Allerton Conference on Communication, Control, and Computing*, 2009.
66. K. Premkumar, Anurag Kumar, and Venugopal V. Veeravalli.

Bayesian quickest transient change detection. In *Proc. Fifth International Workshop on Applied Probability (IWAP)*, 2010.
67. K. Premkumar, V. K. Prasanthi, and Anurag Kumar. Delay optimal event detection on ad hoc wireless sensor networks. *ACM Transactions on Sensor Networks*. Under Second Review.
68. Adi Mallikarjuna V. Reddy, A.V.U. Phani Kumar, D. Janakiram, and G. Ashok Kumar. Wireless sensor network operating systems: a survey. *International Journal of Sensor Networks*, 5(4):236–255(20), Aug 2009.
69. Curt Schurgers, Olivier Aberthorne, and Mani Srivastava. Modulation scaling for Energy Aware Communication Systems. In *Proceedings of the international symposium on Low power electronics and design*, pages 96–99. ACM, 2001.
70. A. N. Shiryaev. *Optimal Stopping Rules*. Springer, New York, 1978.
71. Victor Shnayder, Mark Hempstead, Bor rong Chen, Geoff Werner Allen, and Matt Welsh. Simulating the Power Consumption of Large-Scale Sensor Network Applications. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 188–200. ACM, 2004.
72. Roberto Solis, Vivek S. Borkar, and P. R. Kumar. A new distributed time synchronization algorithm for multihop wireless networks. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 2734–2739, San Diego, December 2006.
73. C. Srisathapornphat and Chien-Chung Shen. Coordinated Power Conservation for Ad hoc Networks. In *Proceedings of IEEE International Conference on Communications*, April 2002.
74. A.G. Tartakovsky and V.V. Veeravalli. Quickest change detection in distributed sensor systems. In *Proc. Sixth International Conference of Information Fusion*, volume 2, pages 75–763, 2003.
75. Crossbow Technology. www.xbow.com.
76. Vlasios Tsiatsis, Scott Zimbeck, and Mani Srivastava. Architecture Strategies for Energy-Efficient Packet Forwarding in Wireless Sensor Networks. In *Proceedings of the International symposium on Low power electronics and design*, pages 92–95. ACM, 2001.
77. Ye Wei, J. Heidemann, and D. Estrin. Medium access control with coordinated adaptive sleeping for wireless sensor networks. *Networking, IEEE/ACM Transactions on*, 12(3):493 – 506, June 2004.
78. Wei Ye, J. Heidemann, and D. Estrin. An energy-efficient mac protocol for wireless sensor networks. pages 1567–1576, 2002.
79. Leena Zacharias and Rajesh Sundaresan. Decentralized sequential change detection using physical layer fusion. *IEEE Transactions on Wireless Communications*, 7(12):4999–5008, Dec. 2008.
80. Z. Zhang, Xuebin Gao, J. Biswas, and J. K. Wu. Moving targets detection and localization in passive infrared sensor networks. In *The 10th International Conference on Information Fusion*, July 2007.