A Low-Complexity Algorithm for Intrusion Detection in a PIR-Based Wireless Sensor Network

Abu Sajana R. [#1], Ramanathan Subramanian [*2], P. Vijay Kumar [#*3], Syam Krishnan [#4], Bharadwaj Amrutur [#5], Jeena Sebastian [#6], Malati Hegde [#7], S. V. R. Anand [#8]

[#] *ECE Department, Indian Institute of Science*
Bangalore, India
{[1] sajana, [3] vijay, [4] syam, [5] amrutur, [6] jeena_sebastian, [7] malati, [8] anand}@ece.iisc.ernet.in

[*] *CSA Department, Indian Institute of Science*
Bangalore, India
[2] sramanathan@csa.iisc.ernet.in

***Abstract*—We present a low-complexity algorithm for intrusion detection in the presence of clutter arising from wind-blown vegetation, using Passive Infra-Red (PIR) sensors in a Wireless Sensor Network (WSN). The algorithm is based on a combination of Haar Transform (HT) and Support-Vector-Machine (SVM) based training and was field tested in a network setting comprising of 15-20 sensing nodes. Also contained in this paper is a closed-form expression for the signal generated by an intruder moving at a constant velocity. It is shown how this expression can be exploited to determine the direction of motion information and the velocity of the intruder from the signals of three well-positioned sensors.**

I. Introduction

In this paper, the challenging problem of intrusion detection using PIR sensors in the presence of clutter arising from wind-blown vegetation is addressed [1]. Manufacturers recommend careful placement of their PIR detectors to prevent false alarms resulting from vegetation, air currents, etc [2]. In [3], [4] and [5], the PIR signal is first high-pass filtered to remove the low frequency components resulting from slow environment changes and then the signal energy is compared against an adaptive threshold. In particular, they use an unsupervised adaptation technique to adjust the energy threshold for target detection. Most other works either use simple thresholding on the PIR signal or thresholding on the energy computed in a window in declaring a detection. In the current paper, we present a low-complexity SVM-training based algorithm that uses the 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 possible. Also SVM makes possible a more fine-grain frequency-domain approach to separating intruder from clutter. The PIR sensor under study in this paper is the analog Panasonic Motion Sensor $AMN24111$. But our results can readily be extended to other 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 WSN, namely the extension of network lifetime. Also contained in the paper is a closed-form expression for the signal generated by an intruder moving at constant velocity. It is shown how this expression can be exploited to determine the direction of motion and the velocity of the intruder from the signals of three well-positioned sensors.

II. Algorithm

We assume that 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 the wind-blown vegetation caused by the wind.

A. Choice of the Sensor

We were initially faced with a choice between the Digital Panasonic Motion Sensor $AMN14121$ and the analog sensor mentioned above. We preferred the analog sensor to the digital PIR sensor as it was easier to distinguish between the spectrum of intruder and clutter from the output of the analog sensor (see Fig. 1).

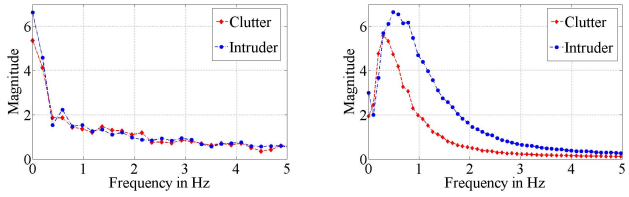


Fig. 1. Spectral signatures of intruder and clutter at the output of the digital and analog PIR sensors respectively.

B. Overview of the Algorithm

For the purpose of maximizing battery life, it was decided to use HT for computing the spectrum of intruder and clutter signals in preference to the computationally more complex 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 HT was preferred as it is less complex compared to WHT, it has ability to reuse past computed HT coefficients for the next window and

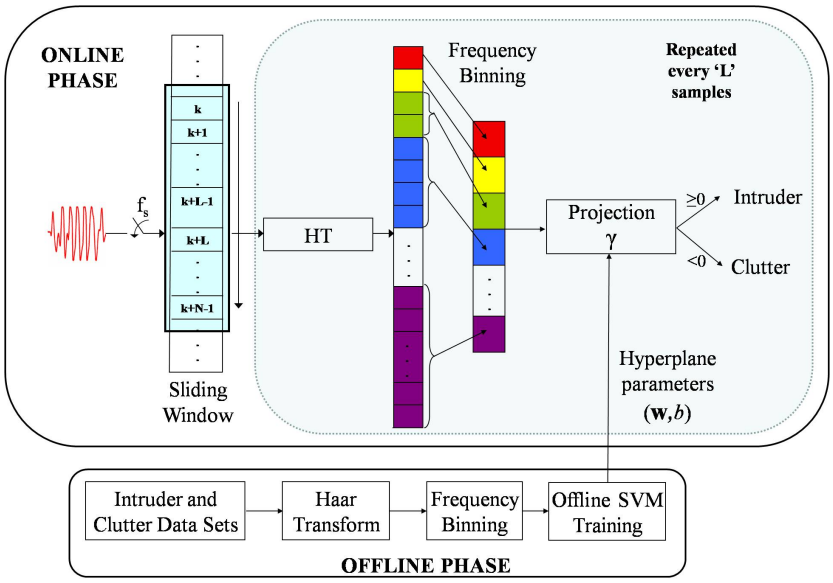


Fig. 2. Functional block diagram of the algorithm.

can potentially be used to yield time-frequency localization information. A sampling frequency (f_s) of $12.5Hz$ was chosen based on the frequency content of intruder and clutter waveforms (see Fig. 1). A functional block diagram of the algorithm appears in Fig. 2. A block of 128 (N) consecutive samples is transformed by 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 offline SVM training) which classifies it as either intruder or clutter. This entire process is repeated every 16 (L) samples.

C. 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, $0 \leq k \leq 6$. 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 these 8 frequency "bins". The Haar signals associated with an example $N = 8$-sample transform are shown in Fig. 3. The time localization information allows reuse of most of the components of the last transformed vector for the current window if there is overlap of the current window with the previous window.

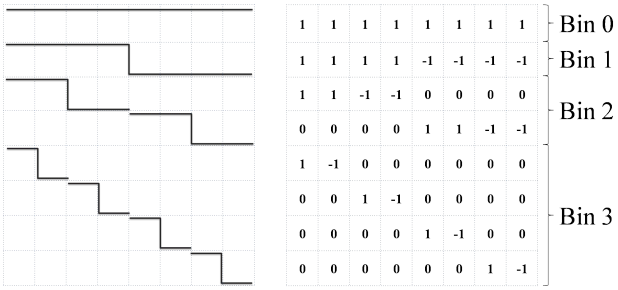


Fig. 3. Frequency bins corresponding to the 8-point Haar matrix.

D. Support Vector Machine

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 [6]. Under linear SVM, the decision surface is chosen by SVM to be the hyperplane in the input space that maximizes the margin, i.e., 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 dimension space which maximizes the margin from points in the larger dimension space to which points in the data set are mapped. Each coordinate in the larger dimension 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. It is to be noted that there is a risk of over-fitting the data with a high degree separating surface that will perfectly separate the training data but it is likely to fail on any new data (see Fig. 4).

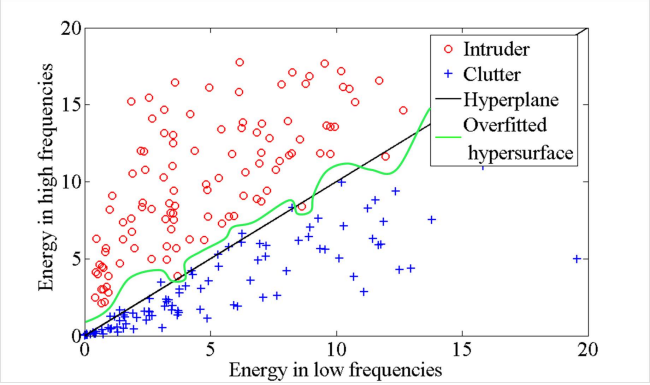


Fig. 4. Illustration of overfitting SVM with higher degree polynomials in $2D$.

E. 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 γ=$\mathbf{w}^T\mathbf{x} + b$ 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. 5. It is to be noted that the computations are in the order of input size 'N'.

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)

Fig. 5. Computational complexity of the algorithm.

F. Training and Testing

The data used for training SVM was collected in a laboratory (i.e., clutter-free) environment (see Fig. 6) in the case of intruder and across 20 outdoor locations on the forested campus of the Indian Institute of Science (IISc) (see Fig. 7) in the case of clutter. The training data set for intruder includes data collected from making a human walk along different straight lines oriented in a variety of ways w.r.t. 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 placed in the close proximity of large fern, or an intruder moving with high velocity) was 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 [7] in MATLAB. Linear SVM, with C set 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. 8 and Fig. 9. Fig. 8 shows four intrusions when the intruder sprinted four times in front of the sensor over a period of 70 seconds, all of which are successfully detected by the algorithm. Fig. 9 shows clutter data collected over a period of 80 seconds, the clutter was successfully rejected.

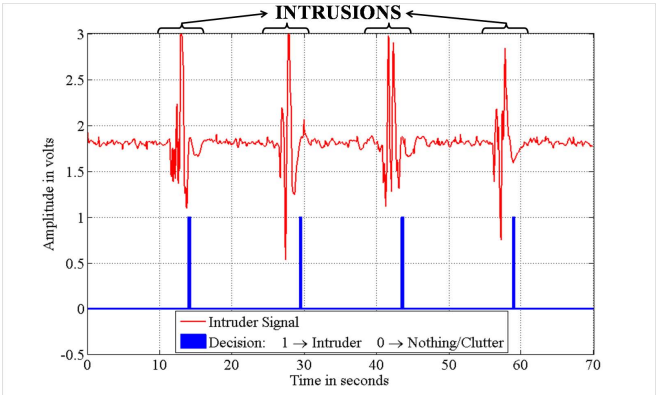


Fig. 8. Linear SVM: Intrusion detected.

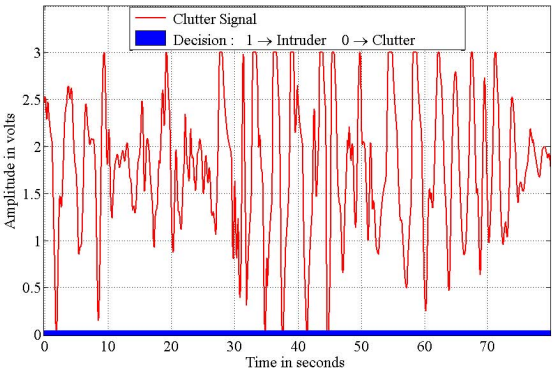


Fig. 9. Linear SVM: Clutter rejected.



Fig. 6. The indoor location used for accumulating intruder data.



Fig. 7. A location in IISc where a part of clutter data was accumulated.

G. 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 noontime in April 2009, at the height of the summer in Bangalore, we observed a significantly larger false alarm rate. Fig. 10 shows a sample waveform recorded in this period. When such summer noontime data was 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, was able to improve 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. We feel that the use of multiple sensors would 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. The next section provides some analysis in this direction. A closed-form expression for a good approximation to the intruder waveform is provided and it is shown how this can be used to determine how to optimally place multiple sensors so as to estimate the path of a human being in motion.

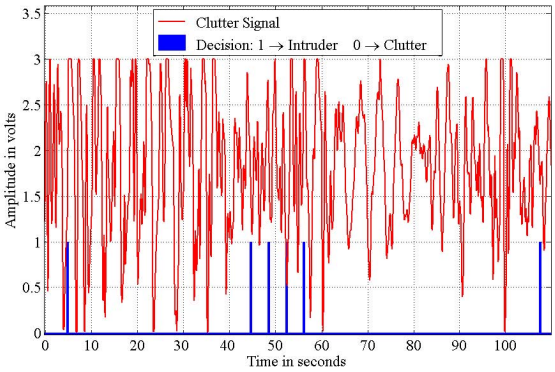


Fig. 10. Quadratic SVM on summer clutter data.

III. Analytical Model for Intruder Signature

The Panasonic PIR sensor has a quad of sensing elements and a multilens which creates a virtual pixel array (VPA) [8] in its field of vision. When an intruder cuts across the virtual beams, the infra-red signal incident on the sensing element can be modeled as a triangular waveform whose period is inversely proportional to the angular velocity ω of the intruder [9]. The sensing element acts upon this infrared signal as a band-pass filter which we will assume filters out all but the fundamental frequency component, whose frequency is thus proportional to ω. Let κ denote the proportionality constant. The intruder is assumed to walk with a uniform velocity along a straight line path in the vicinity of the sensor. This is a reasonable assumption as the sensing range of the sensor is around $6m$.

Let v be the speed of the intruder walking along a straight line making an angle ϕ with the sensor's axis and at a distance of d from the sensor (see Fig. 11). Let 2θ be the horizontal angular coverage of the sensor equal to 110° in for the sensor under study. The sensor responds to the intrusion for $t > 0$. The duration of the signal is limited either by the sensing range or by the angular coverage of the sensor. The instantaneous frequency $f(t)$ of the intruder signature from $\triangle OBC$ is given by,

$$f(t) = \kappa\omega(t) = \kappa\frac{v\cos\psi(t)}{r(t)} = \kappa\frac{vd\sin\phi}{r^2(t)}.$$

Using $\triangle OAB$ we can substitute for $r(t)$ to get

$$f(t) = \frac{\kappa vd\sin\phi}{v^2t^2 + d^2\frac{\sin^2\phi}{\sin^2(\phi+\theta)} + 2vtd\sin\phi\cot(\phi+\theta)} \quad (1)$$

Let $\lambda = \frac{v}{d\sin\phi} = \frac{v}{d_{min}}$ and $\cot(\phi+\theta) = -\lambda t_0$. Thus $0 < \lambda < \infty$ when $v > 0$, $d > 0$ and $0 < \phi < \pi$. Also, d_{min} is the distance of the closest approach of the intruder to the sensor (see Fig. 12). We can now rewrite (1) as

$$f(t) = \frac{\kappa\lambda}{(\lambda(t-t_0))^2 + 1} \quad (2)$$

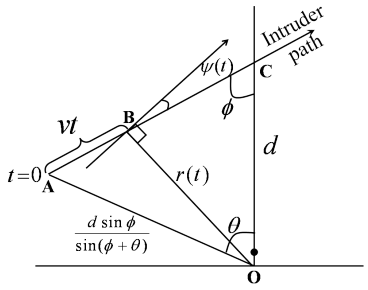


Fig. 11. Geometry used for modeling intruder signature.

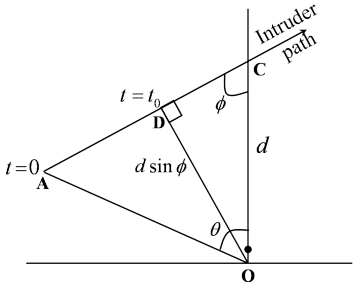


Fig. 12. Illustrating t_0 and d_{min}.

Note that the maximum frequency $f_{max} = \kappa\lambda$ occurs at $t_0 = -\frac{\cot(\phi+\theta)}{\lambda}$. The intruder signature is thus given by,

$$\begin{aligned} s(t) &= \sin\left(2\pi \int_0^t f(t)dt\right) \\ &= \sin\left(2\pi\kappa \arctan\left[\frac{\lambda t}{1-\lambda^2 t_0(t-t_0)}\right]\right) \end{aligned} \quad (3)$$

Fig. 13 compares analytical and actual signals for an intruder signature.

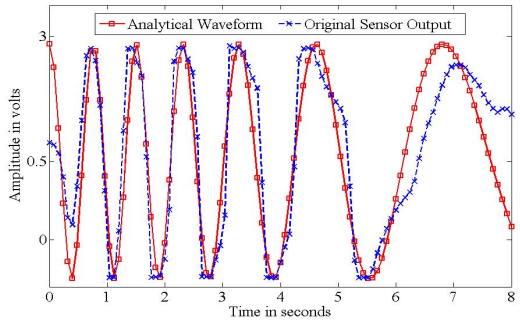


Fig. 13. Intruder signature for $(v, d, \phi) = (0.3, 2, 50°)$.

A. Observations made from the Model

The constant κ corresponds to the density of the beams in the plane of the intruder motion. Hence the analytical expression naturally extends to other differential PIR sensors in general as κ abstracts the lens. λ and t_0 determine the intruder's analytical waveform. λ contains all the information required to track the intruder but λ for different triplets of (v, d, ϕ) can be the same. So to track the intruder, many sensor nodes spaced apart will be required.

IV. TRACKING

The goal in this section is to determine an optimal locationing of multiple sensor nodes, having coordinates (x_i, y_i) respectively, that will enable them to reliably estimate the velocity and direction of motion of the intruder. Set $\eta_i = 1/\lambda_i$. We assume on the basis of (3) that the ith sensor node reliably estimates η_i.

A. Minimum Number of Nodes Required

Let the intruder path equation be $ax + by + c = 0$, which with $r = \frac{\sqrt{a^2+b^2}}{c}$ and $\alpha = \arctan\left(\frac{a}{b}\right)$, can be rewritten as: $xr\sin\alpha + yr\cos\alpha + 1 = 0$. Note that $\eta_i = \frac{d_{\min,i}}{v}$ where $d_{\min,i}$ is the distance from the i^{th} node to the straight-line path of the intruder. With this we obtain the system of equations in the 3 unknowns r, α and v:

$$\eta_i = \frac{\sin\alpha}{v}x_i + \frac{\cos\alpha}{v}y_i + \frac{1}{vr} \quad (4)$$

After some work, it can be shown that 3 sensing nodes will suffice in estimating r, α and v and thus in reliably tracking the intruder.

B. Optimal Positioning of the Sensor Nodes

Tracking involves the transformation: $(\eta_1, \eta_2, \eta_3) \rightarrow (r, \alpha, v)$. The impact of error in the estimates of η_i's on r, α and v should be kept minimum for reliable tracking. Thus, our interest is in maximizing the Jacobian of the transformation carrying out the mapping: $(r, \alpha, v) \rightarrow (\eta_1, \eta_2, \eta_3)$.
Without loss of generality, we assume a coordinate system whose origin is equidistant from the three sensors. Thus we may assume that each sensor is at a constant distance R from the origin. We change variables by writing $x_i = R\cos(\beta_i)$, $y_i = R\sin(\beta_i)$ where $\beta_i = \arctan(\frac{y_i}{x_i})$. With this, (4) becomes

$$\eta_i = \frac{R}{v}\sin(\alpha + \beta_i) + \frac{1}{vr}, \quad 1 \le i \le 3. \quad (5)$$

After some work, it can be shown that the Jacobian is given by

$$\begin{aligned} J &\doteq \begin{vmatrix} \frac{\partial\eta_1}{\partial r} & \frac{\partial\eta_2}{\partial r} & \frac{\partial\eta_3}{\partial r} \\ \frac{\partial\eta_1}{\partial\alpha} & \frac{\partial\eta_2}{\partial\alpha} & \frac{\partial\eta_3}{\partial\alpha} \\ \frac{\partial\eta_1}{\partial v} & \frac{\partial\eta_2}{\partial v} & \frac{\partial\eta_3}{\partial v} \end{vmatrix} \\ &= \frac{R^2}{r^2v^2}\left[\sin(\beta_3-\beta_2) + \sin(\beta_1-\beta_3) + \sin(\beta_2-\beta_1)\right] \end{aligned}$$

The value of J is clearly maximized when $\beta_3 - \beta_2 = \beta_1 - \beta_3 = \beta_2 - \beta_1 = \frac{2\pi}{3}$ and when R is made as large as possible. This suggests that the nodes should be arranged in an equilateral triangle with R as large as possible, subject to the desired node density.

V. Field Testing

Field testing was conducted on the lawns of Electrical Communication Engineering (ECE) Department in IISc campus. In the field tests, three sensors were mounted onto a single node each with an angular spacing of $120°$. When combined with the approximately $110°$ angular field of view of each sensor, this essentially gave each node an omni-directional sensing range (see Fig. 14). Data emanating from the 3 sensors were fed to the 3 ADC channels of the TelosB motes. 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 high speeds, the intruder was in the field of view for only a very short duration. With this in mind, the decision was made to create a double array comprising of two identical, linear and parallel arrays spaced apart by $5m$ (which is just $1m$ under the maximum sensing radius of $6m$). 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 neighbors. 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 sprint at $5m/sec$. There were also no false alarms in the period over which testing was conducted.



Fig. 14. Three sensors platform and the wireless trip-wire.

Acknowledgement

We would like to thank several people for their support and encouragement: Professors V. K. Aatre, A. Kumar, J. Kuri, R. Sundaresan and V. Naik of IISc, Dr. N. Ramamurthy from CAIR, Prof. A. Arora from Ohio State University. Thanks are also due to the WSN team at IISc, in particular, K. Shalini, S. Patil, A. Ranjan, S. Kumar, V. L. Poornima, R. Santhosh and A. Prasad. Finally, thanks are due to Dr. A. V. Krishna from PESIT for introducing SVM to this team.

References

[1] Thomas M. Andrews, "Installation of External Intrusion Detectors in Harsh Environments", in *International Carnahan Conference on Security Technology*, 1993.

[2] *http://en.wikipedia.org/wiki/Passive_infrared_sensor#PIR-based_motion_detector*

[3] Zhiqiang Zhang, Xuebin Gao, Biswas J., Jian Kang Wu, "Moving Targets Detection and Localization in Passive Infrared Sensor Networks", in *The 10th International Conference on Information Fusion*, July 2007.

[4] Lin Gu, Dong Jia, Pascal Vicaire, Ting Yan, Liqian Luo, Ajay Tirumala, Qing Cao, Tian He, John A. Stankovic, Tarek Abdelzaher, Bruce H. Krogh, "Lightweight detection and classification for wireless sensor networks in realistic environments", in *SenSys*, Nov. 2005.

[5] 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, Ken Parker, "Exscal: Elements of an extreme scale wireless sensor network", in *Proc. of the* 11^{th} *IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*, Aug. 2005.

[6] Bernhard E. Boser, Isabelle M. Guyon, 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.

[7] 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.

[8] "MP Motion Sensor (AMN 1,2,4) Data Sheet", Panasonic Electric Works Corporation of America, New Jersey, USA.

[9] *http://www.glolab.com/pirparts/infrared.html*