Energy-Efficient Fault-Tolerant Dynamic Event Region Detection in Wireless Sensor Networks

Hans-Jacob Enemark*, Yue Zhang†, Nicola Dragoni*‡ and Charalampos Orfanidis§
* DTU COMPUTE, Technical University of Denmark
† Shanghai Key Lab for Trustworthy Computing, East China Normal University
‡ Centre for Applied Autonomous Sensor Systems, Örebro University, Sweden
§ Department of Information Technology, Uppsala University, Sweden

***Abstract*—Fault-tolerant event detection is fundamental to wireless sensor network applications. Existing approaches usually adopt neighborhood collaboration for better detection accuracy, while need more energy consumption due to communication. Focusing on energy efficiency, this paper makes an improvement to a hybrid algorithm for dynamic event region detection, such as real-time tracking of chemical leakage regions. Considering the characteristics of the moving away dynamic events, we propose a return back condition for the hybrid algorithm from distributed neighborhood collaboration, in which a node makes its detection decision based on decisions received from its spatial and temporal neighbors, to local non-communicative decision making. The simulation results demonstrate that the improved algorithm does not degrade the detection accuracy of the original algorithm, while it has better energy efficiency with the number of messages exchanged in the network decreased.**

I. Introduction

The Internet of Things envisages a world of uniquely identifiable devices connected through wireless networks. Cheaper devices with connectivity capabilities open up the possibility of designing networks with specific purposes - such as *Wireless Sensor Networks* (WSNs) [1]. WSNs are usually deployed to monitor certain phenomenons in areas that are hard or inconvenient to get to. This could be locations in a big city for monitoring CO_2 levels [2], or the North Atlantic Sea for weather and sea condition reports [3], or even a nuclear reactor for monitoring radiation levels [4].

Due to the inaccessible nature of these WSNs and the unpredictable harsh deployment areas, a WSN has to deal with some problems on its own. One of these issues is the limited energy source. When the energy source is depleted, that node is considered dead. When enough nodes are considered dead, the whole network stops working. To address this issue and prolong the lifetime of the network, *energy efficiency* [5] is a main objective in the overall design of a WSN. Another key issue is that the network is prone to be faulty. Over time sensors may become unreliable and report incorrect readings, or transmissions might be blocked by emerging obstacles, such as birds flying by or a broken antenna. As human service is out of reach for the network, the network itself has to be fault-tolerant [6] [7].Fault-tolerant event detection in WSN applications means to perform proper event reporting under the existence of faults, which might be due to malfunctioning sensor nodes, inaccurate sensor readings or imperfect communication links.

Yue Zhang is corresponding author (yzhang@sei.ecnu.edu.cn)

State of the Art. Neighborhood collaboration and information sharing are very common for distributed fault-tolerant event detection in WSNs to improve detection accuracy, of course, with different assumptions and faults models. Krishnamachari and Iyengar [8] disambiguate sensor readings from events by using sensor readings from neighbors through majority voting and threshold test to make better decisions. Elhadef et al. [9] design a dynamic distributed self-diagnosis protocol Dynamic-DSDP for mobile ad hoc networks based on comparison. They assign tasks to pairs of hosts and compare the outcomes of these tasks. Wang and Cheng [10] design an event region detection approach by fusing local observation and decisions received form neighbors. They use Markov Random Field (MRF) to model the spatial correlation between nodes, and consider space-memory information and local false alarm and detection probabilities under a priori fault model. Ould-Ahmed-Vall et al. [11] take into account different error probabilities of sensor nodes, which are shared with neighbors, and propose an estimator for distributed event detection. Liu et al. [12] design a fault-tolerant event detection scheme for structural health monitoring. They use vectors to send those values extracted from sensor data to the cluster head in their I-FUND to detect nodes with faulty sensor readings. Luo et al. [13] consider energy efficiency due to communication between neighbors and investigate the relationship between detection error and neighborhood size with a given sensor fault probability. Wu and Cheng [14] design an approach for detecting dynamic event region. They adopt dynamic MRF to model the spatiotemporal correlation within local states in an area. With Gaussian noise in sensor readings and characteristics of dynamic events considered, event detection decisions are made by using local decision and decisions received from its spatial and temporal neighbors.

Contribution of the Paper. There is a trade-off between detection accuracy and energy efficiency for fault-tolerant event detection in WSNs. Since communication is the most energy-consuming part of a WSN, it is inevitable to decrease the number of messages exchanged during neighborhood collaboration to achieve better energy efficiency. Making use of the

978-1-4799-8088-8/15/$31.00 ©2015 IEEE

characteristics of dynamic moving away events, this paper focuses on improving an existing fault-tolerant dynamic event region detection approach [14] by designing a switch back mechanism to reduce the number of messages exchanged. Sensor nodes that make decisions by using information received from their spatial and temporal neighbors are switched back to running local decision making algorithm when they are out of the range of the moving away event region.

The remainder of this paper is organized as follows: Section 2 describes the original dynamic event region detection algorithm in [14]; Section 3 proposes an improvement to the above algorithm and gives a theoretical analysis of the improvement; Section 4 evaluates the performance of the improved algorithm by simulation; Section 5 concludes the paper.

II. Original Algorithm

The dynamic event region detection algorithm proposed in [14] is a hybrid one with the following two parts:

1) *CUSUM*: Initially when the WSN is deployed, it is assumed to be in a non-event/healthy state. A non-Bayesian quickest change algorithm called CUSUM (cumulative sum) runs until it detects an event.
2) *P-Algorithm*: After the CUSUM algorithm has detected an event, or a neighbor has indicated the detection of an event, the node switches to run the *P-Algorithm*. In collaboration with neighbors, the node is detecting and in essence tracking the event.

A. Context Information and Observation Model

Before going into the details about the design of the algorithm, it is worth mentioning some context information and observation model, that will impact the design and the later implementation. The context of the algorithm is as follows:

- Square grid structure: All nodes are perceived as being positioned in a square grid with neighbors in the cardinal directions: east, west, north and south, or left, right, up and down.
- Flawless message passing: The lower levels of the protocol stack are assumed to be perfect and a message sent is also assumed to be received with no errors.
- Rare events: The algorithm perceives an event to happen only on rare occasions. This means that even if an event sparks messages being passed around, on average, the time spent on this will be very low.
- Applications: The applications for this algorithm is in tracking of rare events, e.g., a sudden spill of a foreign liquid in a body of a known liquid, such as crude oil in a specific part of an ocean.

Wu an Cheng [14] adopt the following observation model used as input to the main algorithms, in order to mimic the real world sensor readings.

$$y_s^{(n)} = r(H_s^{(n)}) + v_s^{(n)} \tag{1}$$

where $y_s^{(n)}$ is the observed feature value at node s, with $r(H_s^{(n)})$ being the true future value under hypothesis

$$H_s^{(n)} = \begin{cases} 1 & \text{if} \quad \textit{event} \\ -1 & \text{if} \quad \textit{no event} \end{cases} \tag{2}$$

$v_s^{(n)}$ is the observation noise assumed to be Additive White Gaussian Noise with zero mean and variance σ^2, i. e. $v_s^{(n)} \sim \mathcal{N}(0, \sigma^2)$.

B. CUSUM

When deployed, the sensor network is assumed to be without any events. It will remain like this for an unknown period of time n_0, i. e. $H_s^{(n)} = -1$ for $n < n_0$ and $H_s^{(n)} = 1$ for $n \geq n_0$. For each time step, the CUSUM test runs until it detect such a change.

The CUSUM test accumulates log-likelihood ratios of the observations. When the cumulative sum becomes negative, it will be reset to zero. When the sum crosses a threshold, the CUSUM test will stop and send an alarm. Under the above observation model, the log-likelihood ratio of $y_s^{(n)}$ is

$$\begin{aligned} logL(y_s^{(n)}) =& log\frac{p(y_s^{(n)}|H_s^{(n)}=1)}{p(y_s^{(n)}|H_s^{(n)}=-1)} \\ =& \frac{[r(-1)]^2 - [r(1)]^2 + 2y_s^{(n)}[r(1) - r(-1)]}{2\sigma^2} \end{aligned} \tag{3}$$

and the CUSUM test at time n is calculated as follows with $C^{(0)} = 0$,

$$C^{(n)} = max\left(C^{(n-1)} + logL(y_s^{(n)}), 0\right) \tag{4}$$

A binary decision will be made according to the indicator function $\mathbf{1}(.)$ as follows,

$$u^{(n)} = \mathbf{1}\left(C^{(n)} > h\right) \tag{5}$$

The CUSUM test switches to P-Algorithm at time step $n+1$ when the sensor has the binary decision as "1", or when the sensor receives information from its temporal neighbors that an event is detected if the decision is "0" at time step n.

C. P-Algorithm

When the CUSUM algorithm indicates an event has been detected, the sensor node switches into a more active state and initiates the *P-Algorithm*. This algorithm relies on interactions between neighboring nodes in two regards: both spatial and temporal. And the spatiotemporal correlation is modeled as a dynamic Markov Random Field (MRF).

For ease of comprehension, the WSN is assumed to be deployed in a square grid, with unit distance between adjacent nodes. The spatial neighborhood $\mathcal{V}_s^S$ of node s consists of the immediate neighbors in the left, right, up and down directions - i. e. $\mathcal{V}_s^S = \{u, d, l, r\}$. The temporal neighborhood is then $\mathcal{V}_s^T = \{u, d, l, r, s\}$ but with the decisions from time $n-1$ to be used in the decision at sensor s at time n. The neighborhoods are shown in Figure 1.

The main steps in this P-Algorithm are as follows:

A: Gather the results from the neighbors from the previous time step.
B: Approximate the posterior probability of the current state using mean field theory.
C: Repeat step B until convergence.
D: Use the Maximum A Posterior (MAP) rule to make a local decision for the current time step.

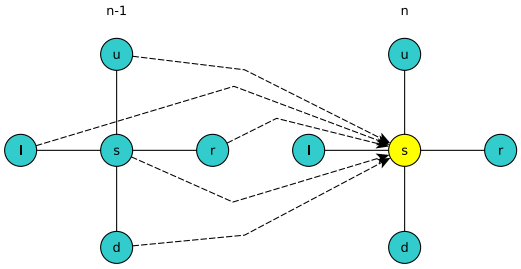


Fig. 1. Visual description of spatial and temporal neighborhoods. Dashed lines are temporal relationships and solid lines are spatial relationships.

According to [14], the Hammersley-Clifford theorem states the equivalence between an MRF and a Gibbs field when defined with respect to a neighborhood system. This leads to the Gibbs distribution

$$P(\mathbf{H}^{(n)}|\mathbf{H}^{(n-1)}) = \frac{1}{Z}e^{-U^{(n)}} \tag{6}$$

and $U^{(n)}$ is the energy function as follows

$$U^{(n)} = \sum_{s\in\mathcal{S}} U_s^{(n)} \tag{7}$$

$$U_s^{(n)} = -\sum_{k\in\mathcal{V}_s^T} \frac{J_{sk}^T}{d_{sk}^2} H_s^{(n)} H_k^{(n-1)} - \frac{1}{2}\sum_{k\in\mathcal{V}_s^S} \frac{J_{sk}^S}{d_{sk}^2} H_s^{(n)} H_k^{(n)} \tag{8}$$

$J_{sk}^T (k \in \mathcal{V}_s^T)$ determines the relationship between the nodes in the temporal neighborhood. J_s^S is similarly describing the relationship between the nodes in the spatial neighborhood. d is the Euclidian distance between the given sensors.

A MAP criterion is used to make the local decision

$$u_s^{(n)} = \max_{H_s^{(n)}\in[1,-1]} P(H_s^{(n)}|y_s^{(n)}) \tag{9}$$

where the posterior probability is found using Bayes' rule as

$$P(H_s^{(n)}|y_s^{(n)}) = \frac{p(y_s^{(n)}|H_s^{(n)})P^P(H_s^{(n)})}{\sum_{H_s(n)} p(y_s^{(n)}|H_s^{(n)})P^P(H_s^{(n)})} \tag{10}$$

where the prior probability $P^P(H_s^{(n)})$ can be expanded as

$$P^P(H_s^{(n)}) = \sum_{\mathbf{H}^{(n-1)}} P(H_s^{(n)}|\mathbf{H}^{(n-1)})P^{post}(\mathbf{H}^{(n-1)}) \tag{11}$$

This shows the Markov property, that the current state of one sensor node $\mathbf{H}^{(n)}$ is only depended upon the previous state of the whole field. However, to find this exact configuration, we would need to know the exact state of the whole field, which would not be traceable. Instead [14] applies mean field theory to approximate the state of the entire field. This is combined with the Gibbs field to get the mean state

$$\begin{aligned}\langle H_s^{(n)}\rangle &= \sum_{H_s^{(n)}} H_s^{(n)} P^P(H_s^{(n)}) \\ &= \sum_{\{H_k^{(n-1)}\}_{k\in\mathcal{V}_s^T}} \frac{1}{\bar{Z}_s} \sum_{H_s^{(n)}} H_s^{(n)} e^{-\bar{U}_s^{(n)}} P^{post}(\{H^{(n-1)}\}_{k\in\mathcal{V}_s^T})\end{aligned} \tag{12}$$

where the energy function

$$\bar{U}_s^{(n)} = -\sum_{k\in\mathcal{V}_s^T} \frac{J_{sk}^T}{d_{sk}^2} H_s^{(n)} H_k^{(n-1)} - \sum_{k\in\mathcal{V}_s^S} \frac{J_{sk}^S}{d_{sk}^2} H_s^{(n)} \langle H_k^{(n)}\rangle \tag{13}$$

with the normalization factor

$$\bar{Z}_s = \sum_{H_s^{(n)}} e^{-\bar{U}_s^{(n)}} \tag{14}$$

The joint probability is simplified as a result of mean field theory,

$$P^{Post}(\{H_k^{(n-1)}\}_{k\in\mathcal{V}_s^T}) = \prod_{k\in\mathcal{V}_s^T} P^{Post}(H_k^{(n-1)}) \tag{15}$$

where $P^{Post}(H_k^{(n-1)})$ is the posterior probability found at each sensor in the temporal neighborhood.

In the P-Algorithm, Equation 12 is updated with the results from neighboring nodes. This is done iteratively until the algorithm has converged when

$$|\langle H_s^{(n)}\rangle^{(I)} - \langle H_s^{(n)}\rangle^{(I-1)}| < \epsilon \tag{16}$$

where I is the number of iterations in the algorithm in that time step. For each iteration, the mean state $\langle H_s^{(n)}\rangle^{(I)}$ is broadcast to the neighbors, in order to refine the neighbors mean state. After convergence, the resulting $\langle H_s^{(n)}\rangle^{(I)}$ is then used for calculating the final prior probability

$$P^P(H_s^{(n)}) = \sum_{\{H_k^{(n-1)}\}_{k\in\mathcal{V}_s^T}} \frac{1}{\bar{Z}_s} e^{-\bar{U}_s^{(n)}} P^{post}(\{H^{(n-1)}\}_{k\in\mathcal{V}_s^T}) \tag{17}$$

used to find the posterior probability - Equation 10 - for the final MAP rule decision.

III. Improved Algorithm

In the original hybrid algorithm, the CUSUM algorithm does not require any communication between nodes. Only when an event is being detected the hybrid algorithm starts tracking the event region with the P-Algorithm. As Wu and Cheng describes the P-Algorithm in [14], they do not mention any stop condition. Without that condition, the P-Algorithm will continue to run indefinitely once started. This is however not feasible if the event, that is being tracked, is no longer present in the vicinity of the given sensor. In the case of a chemical spill in a body of water, the current in the water will most likely have carried the chemical spill away from that sensor. That would mean that the contamination of chemical will not return to that sensor. Hence that sensor has no longer any reason for running the energy consuming P-Algorithm.

A. Improvement

We propose a stop condition for the P-Algorithm, so that unnecessary message passing can be held to a minimum. As the switch from CUSUM to P-Algorithm is initiated by CUSUM detecting an event, so the returning back to CUSUM from P-Algorithm should be the opposite condition, i.e., no event exist any longer.

As seen in Figure 2, the node runs the CUSUM algorithm until it detects an EVENT on its own - (`E_local`) - or it

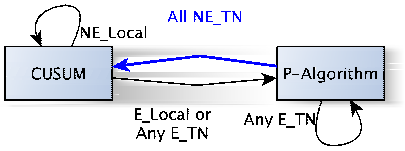


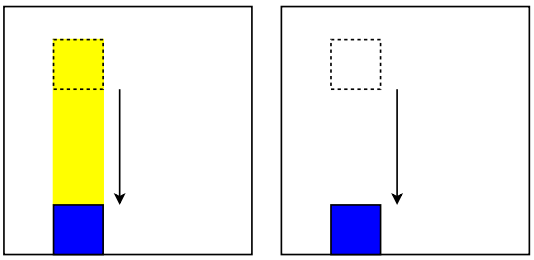
Fig. 2. Modified hybrid algorithm.

gets an EVENT indication message from any of its temporal neighbors - (`Any E_TN`). When that is the case it switches to the P-Algorithm.

Our addition to that structure - which is not mentioned in [14] - is a condition for switching back to CUSUM. As the CUSUM algorithm does not transmit any messages, a transition back to this algorithm will conserve more energy. The mechanism works as follows: when the event has moved away from the sensor and after it has received NO_EVENT indication messages from all of its temporal neighbors, the algorithm goes back to CUSUM. The algorithm detects this condition when none of its temporal neighbors reports an event - (`All NE_TN`). This way it stops sending any unnecessary messages, when there is no event nearby.

B. Analysis

A possible scenario is shown in Figure 3. This shows an event region moving downwards. In the original flow, the event leaves behind a trail of nodes, that still runs P-Algorithm without any event to detect. In the improved flow, the nodes that the event has left switch back to CUSUM and thus stops transmitting messages.



(a) Original flow. (b) Improved flow.

Fig. 3. Example scenario.

As the energy saving is seen as the amount of messages that the WSN is not sending anymore because of the improvement, we need to derive an expression for the amount of messages sent in the WSN. This is described as

$$f(G_P^n) = \mathcal{M} G_P^n \tag{18}$$

$\mathcal{M}$ is the number of message exchanged locally, and G_P^n is the amount of nodes running P-Algorithm. They are the only nodes that are sending messages in this context. For the original algorithm,

$$G_{p,orig}^n = \begin{cases} G_E + \sum_{i=1}^{n} \Delta l_i & \text{for} \quad n \leq T_0 \\ G_E + \sum_{i=1}^{T_0} \Delta l_i & \text{for} \quad n > T_0 \end{cases} \tag{19}$$

T_0 is the time that the event region reaches the boundary. For every time step n until T_0 the event region moves and leaves Δl number of nodes behind, that will continue to run P-Algorithm. These are summed up as time passes. Then at $n > T_0$ no more nodes are switching to the P-Algorithm, and hence G_P^n will stop increasing.

The total number of messages passed around by the original version is illustrated as

$$\sum_{n=1}^{T} G_{P,orig}^n = \sum_{n=1}^{T_0} (G_E + \sum_{i=1}^{n} \Delta l_i) + \sum_{n=T_0+1}^{T} (G_E + \sum_{i=1}^{T_0} \Delta l_i) \tag{20}$$

$$= TG_E + \sum_{n=1}^{T_0} \sum_{i=1}^{n} \Delta l_i + \sum_{n=T_0+1}^{T} \sum_{i=1}^{T_0} \Delta l_i \tag{21}$$

In the improved version, the extra nodes running P-Algorithm outside the event region are removed, and thus

$$G_{p,impr}^n = G_E \tag{22}$$

$$\sum_{n=1}^{T} G_{P,impr}^n = TG_E \tag{23}$$

As an example, Δl_i can be defined as a constant:

$$\Delta l_i = \begin{cases} 0 & \text{if} \quad i = 1 \\ a & \text{if} \quad i > 1 \end{cases} \tag{24}$$

a is the average amount of nodes that the event region leaves at every time step. The theoretical improvement is

$$\frac{\sum_{n=1}^{T} G_{P,impr}^n}{\sum_{n=1}^{T} G_{P,orig}^n} = \frac{G_E}{G_E + \frac{a}{T}(T_0 - 1)\left(T - \frac{T_0}{2}\right)} \tag{25}$$

If we consider a critical time and set $T = T_0$, we get

$$\frac{\sum_{n=1}^{T} G_{P,impr}^n}{\sum_{n=1}^{T} G_{P,orig}^n} = \frac{1}{1 + \frac{a}{2G_E}(T_0 - 1)} \tag{26}$$

The critical time $T = T_0$ is the situation where we do not consider any time steps after the event region has reached the boundary of the WSN deployment area. If the event region - e. g. the pollution in the water - reaches the boundary, it might be out of control and too late to clean up.

This shows that the size of the improvement depends on the size of the event region and the time it takes for the event region to reach the boundary. The size of performance improvement in this context is the amount of messages, that are not sent in the network, because some nodes has switched back to the CUSUM algorithm.

IV. Performance Evaluation

We simulate the above original and improved hybrid algorithms in the simulation framework SimPy v3.0.5 with the programming language Python 3.4.1. The development laptop runs Arch Linux running Linux kernel 3.16.1 64-bit.

We use the same simulation scenario and parameter settings as those in [14], which simulates polluted areas blown by wind to shift downwards (south) and the shapes of the areas are randomly deforming. The size of the simulation is a grid of 100x100 sensor nodes. The parameters for the spatiotemporal model are set as follows: $J_{ss}^T = 6.25$, $J_{su}^T = 5.25$, $J_{sd}^T = J_{sl}^T = J_{sr}^T = 0.25$, $J_{su}^S = J_{sd}^S = J_{sl}^S = J_{sr}^S = 0.25$. Here J_{ss}^T and J_{su}^T have larger values imply the state of a sensor node s at time

n is highly dependent on the states of its upper neighbor and itself at time $n-1$, with the moving downwards phenomenon effectively simulated.

We adopt two metrics for evaluating the improved algorithm. One is error rate, the ratio of the number of sensors making wrong decisions to the total number of sensors. The other is the amount of sensors running P-algorithm.

A. Average Error Rate

The average error rate (AER) is taken for each time step and then averaged over the entire run of 100 time steps. This procedure is done with both the original and the improved implementation. It is compared against Signal to Noise Ratio (SNR). The original data of the AER is given in Table I.

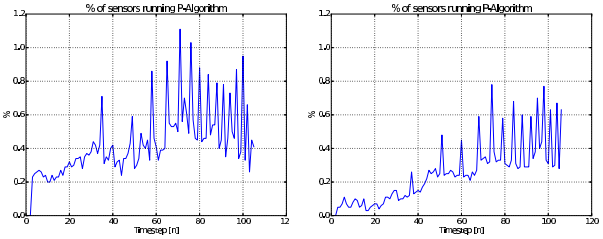
TABLE I
AER VS. SNR

SNR	-15	-10	-5	0	5	10
AER-Orig.	0.4584	0.3593	0.1581	0.0149	0.0144	0.0142
AER-Impr.	0.4602	0.3624	0.1573	0.0150	0.0143	0.0144

This shows two things: 1) the modification to the discussed hybrid algorithm does not seem to affect the AER at all. 2) This implementation is highly sensitive to the noise level as indicated by the AER dropping as the SNR goes up. The SNR sensitivity was however mentioned in the original paper [14], and as such is not a surprise.

B. Amount of Sensors Running P-algorithm

The comparison between Figures 4(a) and 4(b), both of which use an SNR of 10 dB, shows a reduced number of sensor nodes running P-Algorithm in the improved version. This shows the advantage of the added mechanism of the possibility of returning to the CUSUM test.



(a) Original Algorithm. (b) Improved Algorithm.

Fig. 4. The number of sensors running P-Algorithm.

With a reduced number of sensors running the P-Algorithm, the network conserves more energy by reducing the amount of necessary messages sent.

V. CONCLUSION

This paper improves an existing fault-tolerant dynamic event region detection hybrid approach [14] which combines a local non-communicative algorithm with a distributed collaborative algorithm. Moving back to the local state detection is useful once the event in question moves away from a giving node. As this reduces the number of active nodes, this will reduce the amount of messages passed between nodes, making the resulting network more energy efficient. This sort of "idle"-mode can be maintained by nodes that are not near any event region. The improvement made to the original algorithm is designing a switch back mechanism that allows to reduce the number of messages exchanged.

Experimental evaluation shows that the average detection error rate of the improved algorithm is similar to that of the original algorithm while with less messages exchanged. This suggests the valid possibility of using the improved algorithm for dynamic event region tracking and not just detection. The idea of a hybrid algorithm may be used as a framework to improve energy efficiency within dynamic event detection - i. e. one non-communicative algorithm combined with a collaborative distributed algorithm. It is left as future work to investigate those possibilities.

ACKNOWLEDGMENT

This paper has partially been funded by the ProFuN project and by the China Scholarship Council during Yue Zhang's visit at DTU in the context of the IDEA4CPS project and National Natural Science Foundation of China (No. 61361136002).

REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422, 2002.

[2] Y. Liu, X. Mao, Y. He, K. Liu, W. Gong, and J. Wang, "Citysee: not only a wireless sensor network." *IEEE Network*, vol. 27, no. 5, 2013.

[3] L. Pu, Y. Luo, H. Mo, Z. Peng, J.-H. Cui, and Z. Jiang, "Comparing underwater mac protocols in real sea experiment," in *IFIP Networking Conference, 2013*. IEEE, 2013, pp. 1–9.

[4] F. Ding, G. Song, K. Yin, J. Li, and A. Song, "A gps-enabled wireless sensor network for monitoring radioactive materials," *Sensors and Actuators A: Physical*, vol. 155, no. 1, pp. 210 – 215, 2009.

[5] G. Anastasi, M. Conti, M. D. Francesco, and A. Passarella, "Energy conservation in wireless sensor networks: A survey," *Ad Hoc Networks*, vol. 7, no. 3, pp. 537 – 568, 2009.

[6] L. Paradis and Q. Han, "A survey of fault management in wireless sensor networks," *J. Netw. Syst. Manage.*, vol. 15, no. 2, pp. 171–190, Jun. 2007.

[7] A. Mahapatro and P. M. Khilar, "Fault Diagnosis in Wireless Sensor Networks: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 4, pp. 2000–2026, 2013.

[8] B. Krishnamachari and S. Iyengar, "Distributed bayesian algorithms for fault-tolerant event region detection in wireless sensor networks," *Computers, IEEE Transactions on*, vol. 53, no. 3, pp. 241–250, 2004.

[9] M. Elhadef, A. Boukerche, and H. Elkadiki, "A distributed fault identification protocol for wireless and mobile ad hoc networks," *J. Parallel and Distributed Computing*, vol. 68, no. 3, pp. 321 – 335, 2008.

[10] T.-Y. Wang and Q. Cheng, "Collaborative Event-Region and Boundary-Region Detections in Wireless Sensor Networks," *IEEE Transactions on Signal Processing*, vol. 56, no. 6, pp. 2547–2561, Jun. 2008.

[11] E. Ould-Ahmed-Vall, B. H. Ferri, and G. F. Riley, "Distributed Fault-Tolerance for Event Detection Using Heterogeneous Wireless Sensor Networks," *IEEE Transactions on Mobile Computing*, vol. 11, no. 12, pp. 1994–2007, Dec. 2012.

[12] X. Liu, J. Cao, and S. Tang, "Fault tolerant complex event detection in wsns: A case study in structural health monitoring," in *INFOCOM, 2013 Proceedings IEEE*, April 2013, pp. 1384–1392.

[13] X. Luo, M. Dong, and Y. Huang, "On distributed fault-tolerant detection in wireless sensor networks," *Computers, IEEE Transactions on*, vol. 55, no. 1, pp. 58–70, Jan 2006.

[14] T. Wu and Q. Cheng, "Online dynamic event region detection using distributed sensor networks," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 50, no. 1, pp. 393–405, January 2014.