Designing and implementing a criticality-based duty-cycled MAC for low-latency mission-critical surveillance applications

Muhammad Ehsan
LIUPPA, University of Pau
Pau, France 64013
Email: muhammad.ehsan@univ-pau.fr

Congduc Pham
LIUPPA, University of Pau
Pau, France 64013
Email: congduc.pham@univ-pau.fr

***Abstract*—Image sensor node's activity is defined based on the application's criticality level and sentry nodes with faster capture rates have higher probability to detect intrusions and will alert neighbor nodes. At the MAC level, we consider duty-cycled approaches to periodically set nodes in sleep mode for energy preservation. However, in doing so, care must be taken to also preserve the quality of event detection and sentry nodes must still be able to quickly propagate alert messages to meet the strong requirements of mission-critical surveillance applications in communication delays. We propose an original approach to dynamically determine the duty-cycle length of image sensor nodes to increase the probability of matching active period between nodes, thus reducing the alert latency while globally reducing the energy consumption.**

***Index Terms*—Image Sensors; Duty-cycled MAC; Mission-critical; Quality of service.**

I. Introduction

Our research considers mission-critical surveillance applications where image sensor nodes are thrown in mass, randomly, to start the surveillance process, e.g. intrusion/anomaly detection, situation awareness,... When an intrusion is detected by a node it will (i) send a number of images to the sink and (ii) alert a number of neighbor nodes. Alerts could be propagated at k hops but this is not illustrated in the figure. Fig. 1 illustrates the alert process in which neighbor nodes are put in alert mode (red nodes).

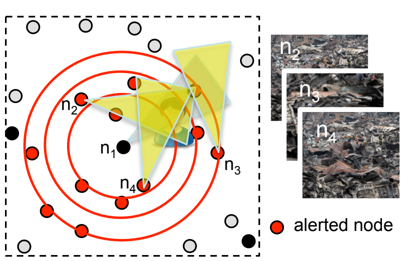

Fig. 1: Alert propagation

We address the Medium Access Control (MAC) layer for providing low energy consumption and low latency for alert propagation. In Fig. 1, it is desirable that neighbor nodes can receive the alert indication as soon as possible in order to propagate the alert towards the sink. However, as event detection in wireless sensor networks (WSN) can be quite sporadic and nodes be idle for a long period of time MAC layers usually adopt a duty-cycled behavior in order to save the energy consumption of maintaining the radio module awake listening for incoming packets: an active or listening period alternates with an inactive or sleep period. A simple approach for duty-cycling such as the one proposed by the 802.15.4 standard can be improved with synchronization features to have common active periods (SMAC [1], TMAC [2] to name a few), with low-power listening (LPL) capabilities and preamble transmissions (B-MAC [3], X-MAC [4], TP-MAC [5] to name a few) or adapted to bursty traffic [6]. Reader can refer to [7] to have a recent survey of MAC protocols for WSN. While synchronous approaches are not scalable for large networks, LPL and preamble-based approaches still suffer from high latencies when node's sleeping period is large. We propose to adapt the active period of sensor nodes to provide low-latency alert communication. In Fig. 1, neighbor nodes for node n_1 will set their listening period according to the criticality level of node n_1: the higher the criticality level, the longer the listening period. Additionally, the node's listening period will also depend on the node's redundancy level in order to determine a listening period which will not compromise the node's lifetime. The contribution in this article is based on a criticality model we developed in [8] for image sensors: (i) each sensor node n has a frame capture rate which depends on the criticality level and node n's redundancy level, then (ii) each neighbor node n_i will set its listening period according to node n's frame capture rate and its own redundancy level. Although our approach has been designed for image sensor the proposition can also work with traditional scalar sensors with disk coverage. Our contribution can also be used with LPL and preamble approaches to determine the receiver periodic channel sampling interval, thus reducing the cost of preambles.

The remainder of the paper is structured as follows. Section II reviews our criticality-based scheduling. Our proposed MAC approach is explained in Section III. Simulations and results are shown in Section IV. The implementation is described in Section V and we conclude in Section VI.

II. CRITICALITY-BASED NODE SCHEDULING

Most of previous works on intrusion detection applications [9], [10], [11], [12], [13] focused on coverage and energy optimizations without explicitly having the application's criticality in the control loop. For instance, with image sensors, the higher the capture rate is, the better relevant events could be detected and identified. However, even in the case of very mission-critical applications, it is not realistic to consider that video nodes should always capture at their maximum rate when in active mode. In randomly deployed sensor networks, provided that the node density is sufficiently high, sensor nodes can be redundant (nodes that monitor the same region) leading to overlaps among the monitored areas. Therefore, a common approach is to define a subset of the deployed nodes to be active while the other nodes can sleep. One obvious way of saving energy is to put in sleep mode nodes whose sensing area are covered by others. However, in mission-critical applications where it is desirable to increase responsiveness, nodes that possess a high redundancy level (their sensing area are covered many times by other nodes so that they have many cover sets) could rather be more active than other nodes with less redundancy level.

In [8] the idea we developed is that when a node has several covers, it can increase its frame capture rate because if it runs out of energy it can be replaced by one of its cover sets. Then, depending on the application's criticality, the frame capture rate of those nodes with large number of cover sets can vary: a low criticality level indicates that the application does not require a high image frame capture rate while a high criticality level does. We proposed to link the criticality level to the number of cover sets by concave and convex curves as illustrated in Fig. 2.

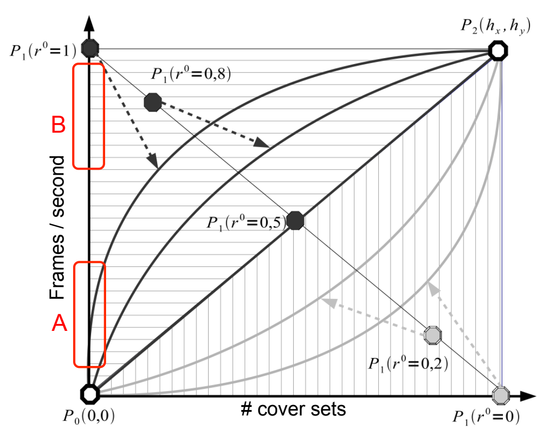

Fig. 2: The Behavior curve functions

These type of curves have the following interesting properties for mission-critical applications:

- a concave curve has most projections of x values on the y-axis close to 0 (Fig. 2 box A). Such curve could represent "low criticality" applications that do not need high frame capture rate;
- a convex curve where most projections of x values on the y-axis are close to the maximum frame capture rate (Fig. 2 box B). Such curve could represent "high criticality" applications that need high frame capture rate;

We proposed in [8] to use a Bezier curve to model the 2 application classes. 3 points can define a convex (high criticality) or concave (low criticality) curve: $P_0(0,0)$ is the origin point, $P_1(b_x, b_y)$ is the behavior point and $P_2(h_x, h_y)$ is the threshold point where h_x is the highest cover cardinality and h_y is the maximum frame capture rate determined by the sensor node hardware capabilities. As illustrated in Fig. 2, by moving the behavior point P_1 inside the rectangle defined by P_0 and P_2, we are able to adjust the curvature of the Bezier curve, therefore adjusting the criticality level: according to the position of point P_1 the Bezier curve can move from a convex to a concave form. P_1 therefore defines a criticality level r^0 which is between 0 and 1, 1 being the highest criticality level which requires fast frame capture rate.

Assuming $P_0(0,0)$, $P_1(b_x, b_y)$ and $P_2(h_x, h_y)$ we can define the Bezier curve (BV) as follows:

$$\begin{array}{rcl} BV : [0, h_x] & \longrightarrow & [0, h_y] \\ X & \longrightarrow & Y \end{array}$$

$$BV_{P_1,P_2}(X) = \begin{cases} \frac{(h_y - 2b_y)}{4b_x^2} X^2 + \frac{b_y}{b_x} X & if\ (h_x - 2b_x = 0) \\ (h_y - 2b_y)(\propto(X))^2 + 2b_y \propto(X), & if\ (h_x - 2b_x \neq 0) \end{cases}$$

$$Where\ \propto(X) = \frac{-b_x + \sqrt{{b_x}^2 - 2b_x * X + h_x * X}}{h_x - 2b_x} \wedge \begin{cases} 0 \leq b_x \leq h_x \\ 0 \leq X \leq h_x \\ h_x > 0 \end{cases}$$

We then define the Rk function such that varying r^0, the dynamic risk level, between 0 and R^0 gives updated positions for P_1 thus obtaining corresponding values for b_x and b_y:

$$\begin{array}{rcl} Rk : [0, R^0] & \longrightarrow & [0, h_x] * [0, h_y] \\ r^0 & \longrightarrow & (b_x, b_y) \end{array}$$

$$Rk(r^0) = \begin{cases} b_x = -h_x \times r^0 + h_x \\ b_y = h_y \times r^0 \end{cases}$$

If we set the maximum cover set cardinality to 12 and the maximum frame capture rate to 3fps we have $P_2(h_x, h_y) = (12, 3)$ and Table I shows the corresponding capture rate for some relevant values of r^0.

TABLE I: Capture rate in fps when P_2 is at (12,3).

r^0	1	2	3	4	5	6	7	8	9	10	11	12
0	.01	.02	.05	0.1	.17	.16	.18	.54	.75	1.1	1.5	3
.1	.07	.15	.15	.17	.51	.67	.86	1.1	1.4	1.7	2.1	3
.4	.17	.15	.55	.75	.97	1.1	1.4	1.7	2.0	2.1	2.6	3
.6	.16	.69	1.0	1.1	1.5	1.8	2.0	2.1	2.4	2.6	2.8	3
.8	.75	1.1	1.6	1.9	2.1	2.1	2.5	2.6	2.7	2.8	2.9	3
1	1.5	1.9	2.1	2.4	2.6	2.7	2.8	2.9	2.9	2.9	2.9	3

With this criticality-based scheduling approach nodes with high number of cover-set will implicitly become sentry nodes by having a higher frame capture rate. The motivation of the work we present in this paper is to allow fast propagation of alert messages from the sentry nodes to the sink. For that purpose, we will propose an adaptive listening period, making each neighbor node of a sentry node ready to receive the alert message and to forward it to the sink. We will describe our contribution in the next sections.

III. Criticality-based Adaptive MAC Protocol Design

In our application scenario a node detecting an event will first send an alert to the sink. Note that any node can detect an event but according to the previously described criticality-based scheduling approach, some nodes will have higher capture rates than others and therefore will act as sentry nodes because they have a higher probability of detecting an intrusion or any changes in the environment (under the assumption that events occur uniformly in the covered area). The node's activity at the application level (the image capture process to detect events) can actually be independent from the radio activity which is the focus of this work. As indicated previously, all nodes have a radio duty-cycled behavior where the radio module is put to sleep for some time, and then waked up to listen for other nodes wanting to communicate with it, e.g. transmission of an alert message for instance. In our scenario, it is desirable that neighbor nodes of a sentry can receive the alert message as soon as possible to (i) increase their criticality level and, most importantly, (ii) propagate the alert towards the sink.

Our contribution works in 2 phases. The first phase is to determine for each image sensor node its associated sentry node, i.e. the image node in its neighborhood with the highest frame capture rate. A node with an associated sentry node will be called a follower node. Then in a second phase, we adapt the follower node's listening period to increase its responsiveness in case of alerts: be ready to receive and quickly relay data to the sink.

A. Sentry selection phase

In this first phase, after having determined its cover-set and frame capture rate [8], every node broadcasts these information. Once all the nodes have finished broadcasting, each node can identify the node with the highest capture rate in its neighborhood. That node is termed as sentry node or master node in its neighborhood. Remind that the capture rate of any node is calculated using the Bezier curve model described previously with examples shown in Table I. Fig. 3 depicts the end of phase 1 where a sentry node (the black node with the highest frame capture rate) has been identified and associated to follower nodes in a given neighborhood.

Once the sentry node has been identified and its frame capture rate known, the second phase is to set the follower node's radio duty-cycling pattern. We propose that the listening period of the follower nodes be calculated in relation to the frame capture rate of their sentry node. However another important factor to consider is the follower node's redundancy level. When a node has several cover sets, if it runs out of energy it can be easily replaced by one of its cover sets. Therefore this follower node can afford to have a duty cycling pattern with longer listening time. This is the purpose of phase 2 described in the next paragraphs in more details.

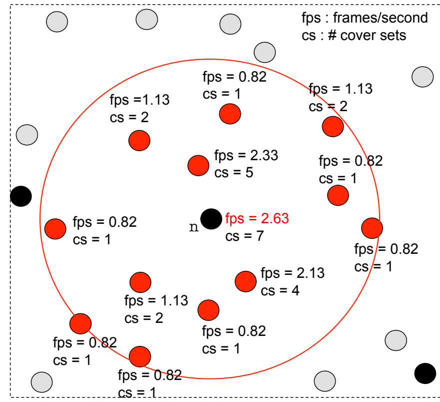

Fig. 3: Sentry node selection at the end of phase 1

B. Determining duty-cycling pattern

If a follower node has a small number of cover sets then it is preferable that it preserves its energy because it can hardly be replaced. This means that each follower node of a given sentry node may have different listening time depending on the size of its cover-sets. We propose that the duty-cycling pattern of a follower node follows the convex/concave model previously described in Fig. 2 in order to maintain the properties of criticality-based scheduling. However, the y axis will now give the corresponding duty-cycle value (between 0 and 1, corresponding to the listening period ratio) based on the cardinality of the cover sets of the follower node itself expressed on the x axis and the sentry node's frame capture rate, see bottom part of Fig. 4.

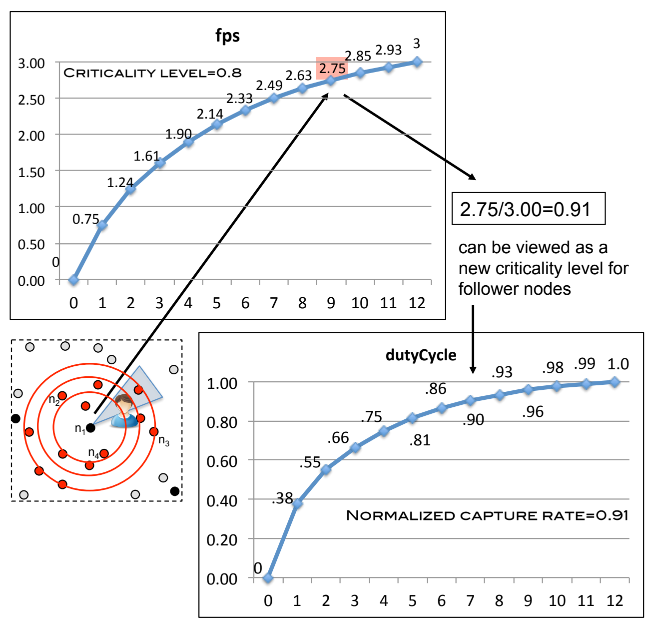

Fig. 4: Criticality curve example

Actually, the sentry node's capture rate value is normalized against the maximum frame capture rate and is used as a new criticality level for the node, whose duty-cycle value is being calculated. In this duty-cycle model, we therefore now have $P_2(h_x, h_y) = (12, 1)$: maximum considered number of cover sets is 12 and duty-cycle ratio is between 0 and 1. The concave curves will represent the smallest capture rate (normalized) where most duty-cycle values will give have smaller listening periods, i.e. values are near to zero, unless if a node has high number of cover sets in which case it will have a larger listening period. Similarly, the convex curves represent the highest capture rate (normalized). In this case the duty-cycle values calculated for the follower nodes will be longer. Follower nodes with larger number of cover sets will have duty-cycle values longer than those with smaller number of cover sets.

Fig. 4 illustrates the entire duty-cycling computation process at follower nodes. In this example, with a criticality level of 0.8 (which gives a convex curve), a node having 9 cover sets will capture at a frame rate of 2.75 fps. Assuming that this node is selected as a sentry node, then its neighbors will use its capture rate to compute their own duty cycle value. Therefore we see in Fig. 4 how the capture rate is normalized (against the maximum capture rate defined by hardware constraints, 3fps in the example) and used as a new criticality level for computing the duty cycle value at a follower node, taking its number of cover sets into account. Here, this new criticality level gives a curve which is more convex, i.e. most values on the y axis will be in the upper half of the curve even with smaller number of cover sets, which means longer duty-cycle values for follower nodes.

As mentioned previously, our approach can also be a very effective method for the preamble length calculation in preamble-based MAC protocols, like B-MAC [3], X-MAC [4]. We will describe in the next section the performance evaluation of our method and comparisons with a static duty-cycle approach.

IV. Simulation Results

To evaluate our approach we conducted a series of simulations using the OMNET++/Castalia simulator. For these set of experiments, we randomly deployed 110 sensor nodes in a 400mx400m area. Each sensor node captures with a given number of frames per second (between 0fps and 3fps). We set the maximum number of cover sets for a node to be 12: nodes with higher number of cover sets will only consider 12 cover sets. Minimum duty cycle is fixed at 0.1 and the criticality level is set at 0.8. Random intrusions are introduced in the simulation model and nodes can detect an intrusion if the intruder is covered by their field of view at the time of the image capture. Upon intrusion detection, a node will broadcast an alert message.

We compared our approach with a traditional static duty-cycled MAC protocol with varied duty cycle values: 0.5, 0.6, 0.7 and 0.8. For instance, a cycle duration of 1s with a duty-cycle value of 0.8 will give 0.8s of radio activity (e.g. can receive) followed by a 0.2s period of inactivity (e.g. can not receive). The follower node's duty cycle values varies depending on its number of cover sets and capture rate of their sentry node. A sentry node does not need to keep its radio active for a long period of time, hence its duty cycle is kept at minimum, i.e. 0.1.

To verify our approach, we designed all the follower nodes to respond with an acknowledgment message on reception of an alert, confirming the reception of the alert. The responses received from the followers confirm that the alert was successfully propagated. On the other hand, no response from the followers goes on to show that the follower nodes were on the sleep mode and they did not received the alert message. Now this means that the alert was sent but none of the follower nodes was available to hear that communication, alert was not propagated and it was not relayed to the sink, which can have severe consequences for applications of critical nature. The total number of alerts sent by all the sentry nodes is approximately 1070. Fig. 5 shows the total number of alerts sent by sentry nodes to which no responses were received, i.e. the number of alerts which were not propagated through the network.

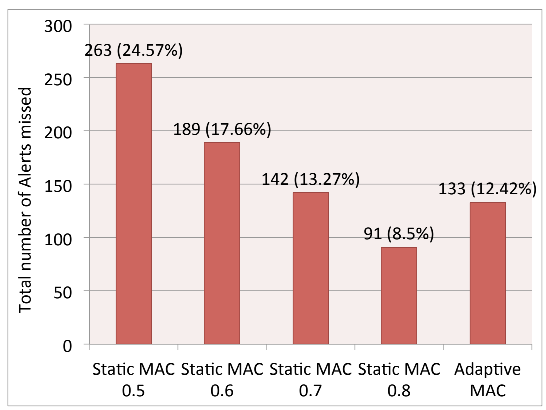

Fig. 5: Number of alerts missed

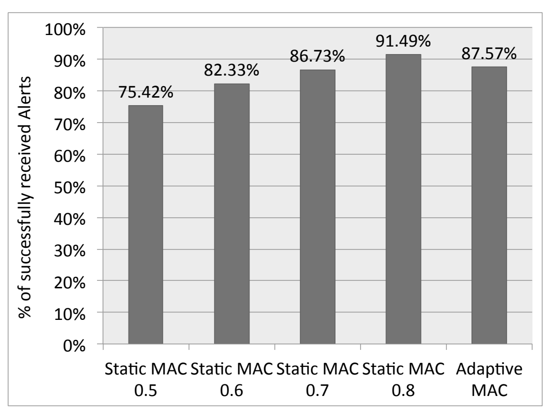

Fig. 6: Received and successfully propagated alert messages

Fig. 6 shows the percentage of received and successfully propagated alert messages. We see in Fig. 5 and 6 that the results of the MAC protocol proposed in this paper, are second only to a static MAC with 80% duty cycle. Criticality adaptive MAC proposed in this paper shows better results in comparison to duty cycled static MAC with duty cycle of 0.7. Actually, as the number of nodes working at 0.7 duty-cycle or above is small, the results shown in Fig. 5 clearly illustrate the benefit of our criticality adaptive MAC approach: fewer nodes working on high duty cycle values but better responsiveness. Non-sentry nodes also sent 353 alerts and in response received 358 acknowledgements.

Fig. 7 shows the comparison of total energy consumption of all the nodes in the network in Joules, when using the energy model of Castalia. In the figure we can see that our criticality adaptive MAC protocol consumed 48% less energy in comparison to a static MAC with duty cycle of 0.8, for a static with duty cycle of 0.7, the energy saved was around 44%, and the corresponding values were 38% and 32% respectively for static MAC 0.6 and static MAC 0.5.

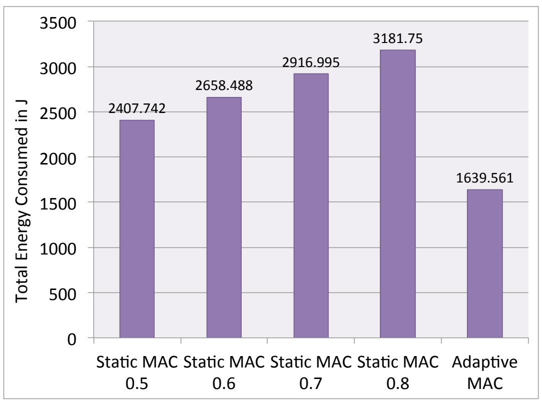

Fig. 7: Comparison of total consumed energy

Taking the global energy consumption of the network, Fig. 8 shows the energy consumed per successfully propagated alert message for the various MAC protocols.

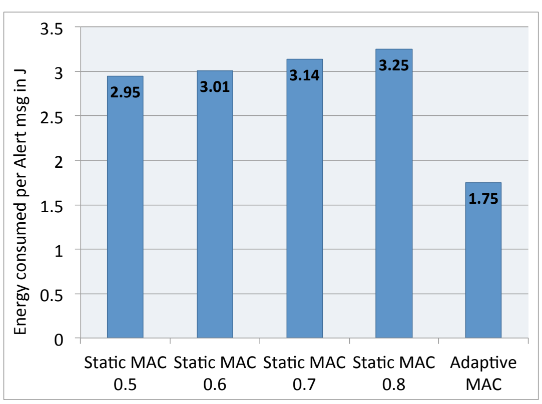

Fig. 8: Comparison of energy consumed per alert message

We see in Fig. 8 that the adaptive MAC approach gives significantly better results in comparison to static MAC with different duty cycle durations. The energy is efficiently utilized to increase the network lifetime.

V. Implementation

A. Follower nodes

We implemented the follower nodes with Libelium Wasp-Mote sensor board (www.libelium.com). The WaspMote is built around an Atmel ATmega1281 micro-controller running at 8MHz. There are 2 UARTs in the WaspMote that serve various purposes, one being to connect the micro-controller to the radio modules. The radio module is an XBee 802.15.4 radio from Digi (www.digi.com) that offers the basic 802.15.4 PHY and MAC layer service set in non-beacon mode. One advantage of the WaspMote is to easily power off completely the radio module from the control software to implement the duty-cycle behavior. The control program waits for initialization messages to define the number of cover-sets (”C” message), the sentry capture rate (”R” message), the cycle length (”T” message) and a static schedule (”D” message) for the static duty-cycling case. When the number of cover-sets and the sentry's capture rate have been received, the criticality-based duty-cycling behavior will start and the follower nodes will wait for alert messages (”A” messages) in order to respond by an ”ACK” message. A led connected to the WaspMote will show when the radio is ON (active) or OFF (sleep), see Fig. 9.

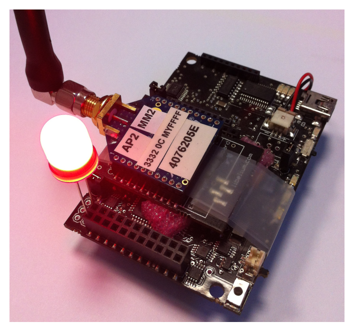

Fig. 9: A Waspmote follower node

B. Sentry node

The sentry node behavior is taken from the simulation model run on a Linux computer: we take the intrusion simulation time trace from a specific sentry node in order to have the intrusion time sequence. A python script reads the time sequence and broadcasts through an IEEE 802.15.4 XBee gateway the corresponding ”A” message at the appropriate moment. For the moment, the simulation is run first, then the time trace is extracted.

C. Test-bed

Fig. 10 shows our test-bed with 1 sentry node (the Linux machine with the XBee gateway) and 5 follower nodes. A shell script will automatically configure each follower node with an appropriate cover-set size by sending a ”C” message and will also broadcast the sentry's capture rate with an ”R” message. On Fig. 10, each follower node is identified by its MAC address (prefixed by 0x0013A200) and has an associated cover-set size. When all follower nodes are configured they start the duty-cycling behavior. We set the cycle length to 3000ms and the frame capture rate of the sentry node to 2.33fps (i.e. the sentry node has 6 cover-sets and run at a criticality level of 0.8, see Fig. 4). Fig. 10 shows a snapshot where each follower node has computed its duty-cycle ratio and where 2 follower nodes have their radio ON, 408BC823 and 4086D828 identified by the green ”active” banner, while the 3 others are in sleep mode.

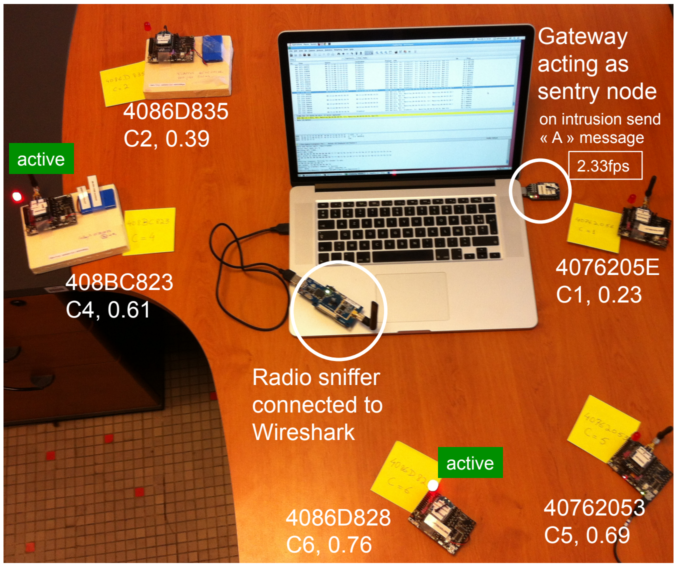

Fig. 10: A Waspmote follower node

D. Preliminary results

Fig. 10 also shows a radio promiscuous sniffer plugged into the `wireshark` packet analysis tool to record all the exchanged messages. We made the sentry node script stop after 1000 intrusions (we therefore have 1000 ”A” messages) and we counted the number of ”ACK” messages from followers nodes. In the best case, when all follower nodes are active at the time of the alert message, we would have a total of 5000 ACK messages. We compared our criticality-based duty-cycle MAC approach with a static duty-cycle behavior of 0.5, 0.6, 0.7 and 0.8 as previously. A given alert message failed to be propagated if there are no ACK messages for this alert message. The `wireshark` trace allows us to easily track down such cases and we experimentally found results close to those presented in Fig. 5.

E. Energy consideration

We also measured the energy consumption of the WaspMote node when the radio is powered-off and when the radio is always powered-on. With the radio OFF and minimum processing tasks, the WaspMote consumes about 0.036J/s (36mW). With the radio ON and ready to receive and propagate alert messages it consumes about 0.236J/s (236mW). With a static duty-cycle strategy the energy gain can be directly obtained from the duty-cyle ratio as all nodes have the same ratio. With our criticality-based duty-cycle approach we can still know the overall energy gain with the duty-cycle value of each follower node.

VI. Conclusions

In this paper, we proposed a duty-cycled MAC protocol for low latency alert propagation and low energy consumption. In a network all the nodes work for the same purpose so if a node has high redundancy, it can be used more extensively for detection purposes. The key point in our approach is to link the duty cycle of nodes with image capture rate and number of cover sets.

Simulations have shown the efficacy of our approach. The results have shown that our approach was responsive to high number of alerts in comparison to various duty cycle lengths for a static duty-cycle MAC. At the same time the energy consumed for the whole network was minimum for our approach, which are very promising results. We implemented our approach and experimental measures have confirmed our simulation results. In future we want to extend our sentry node selection to two-hops and we want the duty cycle of nodes on the route to sink to be calculated with this approach, to receive images at the sink with minimum latency.

Acknowledgment

This work is partially supported by the Aquitaine-Aragon OMNIDATA project.

References

[1] W. Ye, J. Heidemann, and D. Estrin, “Medium access control with coordinated adaptive sleeping for wireless sensor networks,” *IEEE/ACM Trans. Netw.*, vol. 12, no. 3, pp. 493–506, Jun. 2004.
[2] T. Van Dam and K. Langendoen, “An adaptive energy-efficient mac protocol for wireless sensor networks,” in *ACM Sensys*, 2003.
[3] J. Polastre, J. Hill, and D. Culler, “Versatile low power media access for wireless sensor networks,” in *ACM Sensys*, 2004.
[4] M. Buettner, G. V. Yee, E. Anderson, and R. Han, “X-mac: a short preamble mac protocol for duty-cycled wireless sensor networks,” in *ACM Sensys*, 2006.
[5] A. Grilo, M. Macedo, and M. Nunes, “An energy-efficient low-latency multi-sink mac protocol for alarm-driven wireless sensor networks,” in *third international EURO-NGI network of excellence conference on Wireless systems and mobility in next generation internet*, 2007.
[6] S. Ray, I. Demirkol, and W. Heinzelman, “Atma: Advertisement-based tdma protocol for bursty traffic in wireless sensor networks,” in *IEEE GlobeCom*, 2010.
[7] A. Bachir, M. Dohler, T. Watteyne, and K. K. Leung, “Mac essentials for wireless sensor networks,” *IEEE Communications Surveys and Tutorials*, vol. 12, no. 2, pp. 222–248, 2010.
[8] A. Makhoul, R. Saadi, and C. Pham, “Risk management in intrusion detection applications with wireless video sensor networks,” in *IEEE WCNC*, 2010.
[9] O. Dousse, C. Tavoularis, and P. Thiran, “Delay of intrusion detection in wireless sensor networks,” in *ACM MobiHoc*, 2006.
[10] A. Czarlinska and D. Kundur, “Wireless image sensor networks: event acquisition in attack-prone and uncertain environments,” *Multidimensional Syst. Signal Process.*, vol. 20, 2009.
[11] E. Freitas et al. “Evaluation of coordination strategies for heterogeneous sensor networks aiming at surveillance applications,” in *IEEE Sensors*, 2009.
[12] M. Alaei and J. M. J. M. Barcelo-Ordinas, “Priority-based node selection and scheduling for wireless multimedia sensor networks,” in *IEEE WiMob*, 2010.
[13] S. Paniga, L. Borsani, A. Redondi, M. Tagliasacchi, and M. Cesana, “Experimental evaluation of a video streaming system for wireless multimedia sensor networks,” in *10th IEEE/IFIP Med-Hoc-Net*, 2011.