IET Wireless Sensor Systems

Case Study

IET Journals
The Institution of Engineering and Technology

Industrial Internet of things at work: a case study analysis in robotic-aided environmental monitoring

ISSN 2043-6386
Received on 16th March 2017
Revised 11th May 2017
Accepted on 19th June 2017
E-First on 1st August 2017
doi: 10.1049/iet-wss.2017.0032
www.ietdl.org

Vito Scilimati[1], Antonio Petitti[2], Pietro Boccadoro[1], Roberto Colella[2], Donato Di Paola[2], Annalisa Milella[2], Luigi Alfredo Grieco[1] ✉

[1]*Department of Electrical and Information Engineering (DEI), Politecnico di Bari, Bari, Italy*
[2]*Institute for Intelligent Systems for Automation, National Research Council, Bari, Italy*
✉ *E-mail: alfredo.grieco@poliba.it*

Abstract: Nowadays, Internet of things (IoT) and robotic systems are key drivers of technological innovation trends. Leveraging the advantages of both technologies, IoT-aided robotic systems can disclose a disruptive potential of opportunities The present contribution provides an experimental analysis of an IoT-aided robotic system for environmental monitoring. To this end, an experimental testbed has been developed. It is composed of: (i) an IoT device connected to (ii) an unmanned aerial vehicle (UAV) which executes a patrolling mission within a specified area, where (iii) an IoT network has been deployed to sense environmental data. An extensive experimental campaign has been carried out to scavenge pros and cons of adopted technologies. The key results of the authors analysis show that: (i) the UAV does not incur any significant overhead due to onboard IoT equipment and (ii) the overall quality of service expressed in terms of network joining time, data retrieval delay and packet loss ratio satisfies the mission requirements. These results enable further development in larger-scale environment.

1 Introduction

Internet of things (IoT) is a well recognised technology that plays a key role in current innovation trends [1–3]. Generally speaking, IoT envisions a capillary deployment of networked smart devices, which can serve different application domains such as healthcare, manufacturing, transportation, military, smart grids, smart retail, environmental monitoring, agriculture and logistics [2–4].

With reference to connectivity issues, low-power and short range wireless communication technologies can be effectively used in many IoT scenarios. In this context, the IEEE 802.15.4 standard [5] is considered as a leading solution, thanks to its flexibility and energy efficiency. To broaden its scope toward industrial scenarios, time slotted channel hopping (TSCH) multiple access has been added since 2012 to the basic carrier sense multiple access with collision avoidance mechanism [3].

With TSCH, frequency hopping and time-division multiple access (TDMA) mechanisms are used [6]. Frequency hopping consists of changing transmission channel at regular intervals with a pseudo-random algorithm. This improves radio links reliability, thus allowing the provisioning of deterministic IoT connectivity also in noisy industrial settings. TDMA foresees time division into slots assigned to sensor nodes according to a shared schedule. The schedule drives transmission and reception activities over available time slots and frequency channels. Radio-frequency transceivers on top of IoT devices are coherently switched on and off, thus allowing optimal duty cycling policies. As a consequence, ultra-low-power operations are achieved.

In parallel to the IoT development, scientific progress also boosted automation and robots, as they are increasingly present in everyone's daily life in the near future. Robots recently gained momentum in several contexts including healthcare [7, 8], search-and-rescue [9], service robotics [10], manufacturing [11], agriculture [12] and environmental monitoring [13], to name a few.

Nowadays, a mobile robot is able to move autonomously and interact with the environment (e.g. a self-driving car is a special case of a mobile robot [14]). This is particularly true in structured situations [15]. More efforts are still required in big, harsh and dynamic scenarios such as deep ocean waters [16, 17], polar regions [18] and space exploration [19]. Something similar happens in less extended, but not less complex, scenarios such as the human body [20]. Currently, the robotic community is mainly focused on the study of robotic networks. Generally speaking, a robotic network is composed of several robots able to exchange information over a communication network to accomplish collaborative tasks [21].

Researches in IoT-aided robotic systems are paving the way to a digital eco-system, where robots and IoT devices interact on a cooperative basis [22]. This ambitious goal is made possible as most of modern robots are already equipped with sensing, computing and communication capabilities that allow them to execute complex and coordinated operations.

This work explores the interaction between robotic systems and IoT technologies by proposing an experimental analysis of an IoT-aided robotic system for environmental monitoring. In particular, an experimental testbed has been set up. It is composed of: (i) an unmanned aerial vehicle (UAV), equipped with (ii) an IoT device, patrolling an area in which (iii) an IoT network has been deployed.

The UAV is able to follow a mission plan, flying through the area of interest, join the IoT network and effectively retrieve environmental data coming from the sensors belonging to the end nodes. Encouraging results come from low-power consumption overhead contributed by the IoT device onboard of the UAV which is fully compatible with the flight range of the UAV. Moreover, measured quality of service (QoS) indices, i.e. task execution times and packet loss ratio (PLR), resulted in suitable values for the envisioned application.

The rest of this paper is organised as follows. Section 2 provides a complete description of both the robotic networks and Internet Protocol version 6 (IPv6) over the TSCH mode of IEEE 802.15.4 (6TiSCH) technology, with a detailed overview of state of the art of the interaction between IoT technologies and robotic systems. Section 3 presents the operating scenario, the enabling technologies and the way they are integrated. Section 4 describes the envisioned experiments and related results. With Section 5, a complete outline of the outcomes is given, highlighting achievements and proposing future work possibilities.

2 Essential background

The scientific background of this work pertains to Robotics, 6TiSCH Technology and their interaction, which are briefly summarised in this section.

2.1 Robotic systems

In the latest years, robotic systems have been widely adopted in several structured and unstructured contexts. Groups of mobile robots are commonly used in automated warehouses, where a fleet of robots handle the routing of goods among the deposit, the assembly line and the shipping containers. Companies that handle huge amounts of orders such as Amazon and Alibaba make an extensive use of this kind of automation [15]. Robotic technologies are increasingly being used in agriculture to increase the automation level of agricultural machinery [12]. In all these contexts, robots hold the promise to dramatically increase productivity, as they can work night and day with little or no pauses and efficiency initiatives such as resource management and just in time production can be performed. Robotic systems can be really effective for search-and-rescue applications. Aerial Vehicles can help in the mapping process [23], providing the system with a different perspective of the environment and covering zones unreachable by wheeled robots, while ground vehicles equipped with grasping end-effectors can handle environment manipulation [10]. In [8], for example, a novel strategy for the online planning of optimal motion-paths for a team of autonomous ground robots engaged in wilderness search-and-rescue is presented and compared with an alternative non-probabilistic approach. Furthermore, in [11] it is possible to find a comprehensive review about research and applications on a range of topics of importance for implementing mobile robots and automated guided vehicles in manufacturing. These topics include planning, navigation, vehicle localisation and interactions between mobile robots and humans and between groups of mobile robots. In [13], the significant advancements and applications of marine, terrestrial and airborne robotic systems developed for environmental monitoring during the last two decades are collated and discussed.

Since multiple robots outperform single ones in many application domains, the attention of researchers toward robotic networks has grown in the latest years. They are robust, as they eliminate a single point of failure and increase efficiency when a complex task can be split in multiple simpler sub-tasks, thus enabling optimisation procedures at the time of sub-task allocation to agents.

2.2 6TiSCH technology

In addition to classic IoT demands, industrial IoT applications require reliable and low latency communication protocols [24]. The 6TiSCH technology is now configured as a complete framework; thanks to several Internet engineering task force (IETF) working groups (WGs) that focused their attention on the different levels of the protocol stack. For instance, IPv6 over low-power wireless personal area networks (6LoWPAN) [25], routing over low power and lossy networks (LNs) (ROLL) [26] and Constrained RESTful Environments [27] WGs defined specific solutions for each layer [3, 28]. Also based on such results, the IETF 6TiSCH WG, managed to integrate IPv6 and the TSCH mode [29] of the IEEE802.15.4 standard [5].

At the application layer of the 6TiSCH stack, the constrained application protocol (CoAP) protocol [30] is used to grant interoperability. It implements a leave asynchronous request–response protocol over user datagram protocol [31] and translates to hypertext transfer protocol for integration with the web. It allows multicast support and low overhead.

At the network layer, routing protocol (RPL) for low-power and LNs has been proposed by the ROLL WG [26]. This gradient-based distance-vector routing protocol is designed to ease low-power LNs (LLNs)'s formation and management and it organises the network topology as a directed acyclic graph (DAG), possibly partitioned into destination oriented DAGs subject to the optimisation criteria defined within objective functions.

The adoption of the IPv6 protocol on top of an LLN is not straightforward, e.g. the physical layer (PHY) of IEEE802.15.4 is 127 B long, whereas the default IPv6 maximum transmission unit is 1280 B long with a 40 B header. Therefore, IPv6 over networks of resource-constrained nodes (6lo) [25] proposes a standardised approach, aimed at optimising communications, based on link layer features introduced by 6LoWPAN [6]. This approach is also integrated in the 6TiSCH architecture.

At the medium access control (MAC) layer, the TSCH mode of the IEEE802.15.4 standard is used [5, 6, 24, 32]. Thanks to time synchronisation and CH, it provides almost deterministic access to the medium thus enabling high reliability and maintaining low-duty cycles, lowering radio chip activities; therefore, power consumption. CH strategy mitigates the effects of interferences and multi-path fading. TSCH mode foresees synchronisation based on a repeating sequence of time slots, namely slotframe. A time slot, indeed, represents the amount of time reserved for a single operation to complete. In a slot, devices can either transmit, receive or keep their radio off.

Finally, IEEE802.15.4 PHY is used at the lowest layer. It represents a healthy trade-off between energy efficiency, range and data rate. It defines 16 radio channels in the industrial, scientific and medical band ranging from 2.4 to 2.485 GHz, together with direct sequence spread spectrum modulation format and signal encoding.

2.3 IoT-aided robotics

Cooperation between IoT and robotic systems is a hot research topic. One possible approach is the so-called Internet of robotic things (IoRT). The IoRT can be seen as an advanced version of cloud networked robots [33]. In IoRT, robotic systems are able to connect, exchange information and share computation resources, context information and environmental data with each other, by means of sophisticated architectural frameworks [34]. Some works dealing with IoRT have been published in recent years [35, 36]. In [35], a control strategy based on a neural network is proposed for the connectivity maintenance and the coverage among multiple IoRT systems, whereas the goal of the work in [36] is maximising the coverage while keeping it uniform. Specifically, a distributed control strategy based on simple motion coordination schema is employed. Both works [35, 36] evaluate the performance of the proposed strategies in a simulated scenario. Another approach for cooperation between IoT and robotic systems is the so-called IoT-aided robotics. In this approach, IoT devices and robots are designed to collaborate in order to reach a common goal. A comprehensive state of the art on the main application domains of IoT-aided robotics is given in [22]. Research efforts have been performed in this direction [2, 37–39], though much remains to be done. In [2], robotic systems are deeply investigated as active part in localisation of nodes in IoT networks as long as detecting and reacting to sensor failures and aggregating sensor data. Contrariwise, IoT networks can effectively be involved in robots localisation, path planning, mapping and sensing procedures. In [37], a robot sensor network is used for wireless communication for remote control, mission cooperation and to report sensed information to the control and reporting centre. The work in [38], instead, shows how simple functionalities such as robot task allocation and robot task fulfilment can benefit from the cooperation between robots and sensors. In [39], the use of AVs for data collection is proved, through numerical simulations, to be more energy efficient than multi-hop data aggregation. In this paper, IoT-aided robotics for environmental monitoring is addressed and an experimental analysis of a real scenario is proposed.

3 Proposed case study

The proposed case study is based on the idea of testing the interactions between IoT devices and robotic systems, to realise a fully working IoT-aided robotic system. In particular, a robot is involved in patrolling an area of interest in which a network made of IoT devices – also referred to as motes – provides updated information retrieved by the end nodes. The test environment is

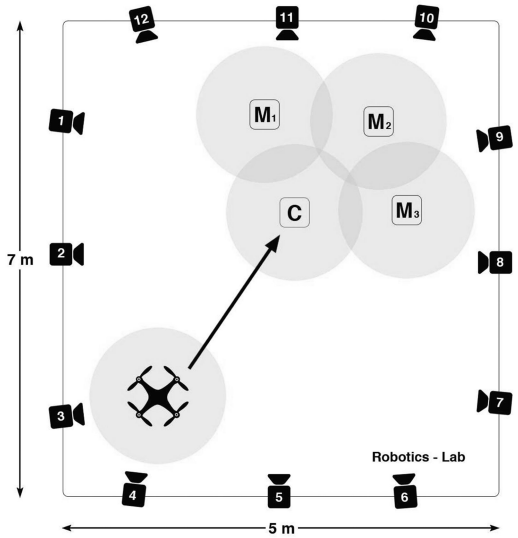

Fig. 1 *Operative scenario. Specifically, 12 cameras composing the MCS, one UAV, three motes (M1, M2 and M3) and one network coordinator (C)*

Table 1 Hardware components of the envisioned testbed

Component	Item	Description	Quantity
IoT network	coordinator	TelosB	1
IoT network	coordinator	Raspberry Pi 2	1
IoT network	child	TelosB	3
IoT network	special node	TelosB	1
MCS	infrared cameras	Vicon Bonita 10	12
MCS	workstation	Hp Z440	1
UAV	quadcopter	EMAX Nighthawk 250	1
UAV	3DRobotics Pixhawk	3DRobotics Pixhawk	1
UAV	onboard computer	ODROID XU-4	1

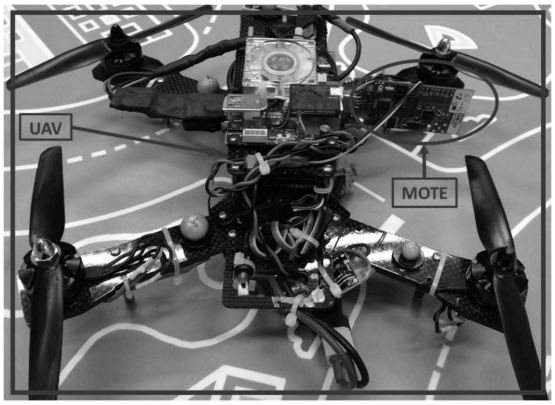

Fig. 2 *UAV equipped with the mote (TelosB)*

depicted in Fig. 1. It consists of a laboratory of about 35 m^2, an area that allows the small-sized robot to move freely and unhindered. Objects moving inside the volume of the laboratory are detected and tracked by a motion capture system (MCS). The transmission power of wireless equipment has been sized according to the scale of the laboratory. Overall, the test environment has been organised in such a way that it reproduces, in a downscaled version, a typical scenario, where IoT and robotic system integration would be meaningful to take place such as agricultural field monitoring or search-and-rescue missions. Fig. 1 represents the operative scenario. Here, several entities can be identified:

- the MCS, which uses 12 sensing elements;
- the network coordinator, which processes requests from robot operating system (ROS);
- the UAV node; and
- the child nodes of the IoT network. They are deployed over the area of interest and used to communicate sensor data.

3.1 Hardware

Developing an interaction between IoT devices and robotic systems requires dedicated hardware solutions that allow the creation of applications and services supporting different IoT contexts such as environmental protection, logistics, healthcare and search-and-rescue scenarios. In this work, a robot has to communicate with the network, and specifically with the network coordinator. To this aim, the robotic unit is equipped with an IoT device, thus becoming a network node. Consequently, information exchange between the network and the UAV is enabled. All the hardware components described here are reported in Table 1.

The IoT hardware platform used to develop the IoT network is one of the most tested and used IoT solutions, Telos rev B board [http://www.memsic.com/userfiles/Datasheets/WSN/telosb_datasheet.pdf], widely referred to as TelosB. It is equipped with a first generation Texas Instruments MSP430 micro-controller unit (MCU) and an IEEE802.15.4-compliant CC2420 radio chip. It also features temperature, humidity and light sensors and it is supplied by a couple of 1.5 V, 2500 mAh AA batteries that enable the mote to work for no longer than 2 days [This value strongly depends on radio activities frequency.].

The root node is connected to a powerful and fully functional elaboration unit, Raspberry Pi 2 Model B [http://www.raspberrypi.org/products/raspberry-pi-2-model-b/]. It is equipped with a 900 MHz quad-core ARM Cortex A7 central processing unit, 1 GB of Double Data Rate type three (DDR3) random access memory (RAM) and four universal serial bus (USB) ports.

The robot is a small custom quadcopter (frame dimensions: 260 mm) developed for indoor research applications. It has a 3DRobotics Pixhawk flight control unit for attitude stabilisation, data gathering from inner sensors and data fusion. Pixhawk is equipped with uBLox global position system, Pixhawk inertial measurement unit, compass and a power module with a stable power supply. An onboard ODROID XU4 ARM computer [http://magazine.odroid.com/odroid-xu4/] running ROS [http://www.ros.org] provides high-level custom tasks such as control and computer vision algorithms. Its carbon fibre frame also holds four MT2204 AGM motors, all powered by a 11.1 V, 2500 mAh battery pack. The setup described is depicted in Fig. 2.

Objects moving inside the area of interest are observed by an MCS composed of Vicon Bonita B10 cameras [http://www.vicon.com/products/camera-systems/bonita/], used for mapping and surveying. These feature an RJ45 Gigabit Ethernet interface and 68 infrared light-emitting diodes and are working at 250 fps frame rate, 1 megapixel (1024 × 1024) resolution and lens operative range up to 13 m.

3.2 Software

The complete software suite used for the experimental testbed is mainly composed of two different parts: ROS and OpenWSN.

ROS is the open source reference OS for robots. It provides a number of features, libraries, drivers, visualisation tools, message passing and data analysis tools, all meant for developing purposes. It is used for two-dimensional (2D) and 3D mapping applications (i.e. objects identification, control and planning in both industrial and research fields) written in several programming languages such as C++ and Python.

IoT devices have been configured using OpenWSN, a well-known open source implementation of the IEEE/IETF 6TiSCH protocol stack [40]. Its structure is made of two main building blocks: the software, namely OpenVisualizer, mainly written in Python, and the firmware, mainly written in C. The former provides real-time network monitoring tools and gateway functionalities, whereas the latter consists of a layered protocol stack that allows constrained devices to wirelessly connect each other and to the public Internet. It also features all the necessary drivers and libraries to support a large variety of boards and

Table 2 Software components of the envisioned testbed

Component	Type	Name
OS for robots	software	ROS
OS for IoT networks	firmware	OpenWSN
OS for IoT networks	software	OpenWSN
multi-thread software	software	Python script

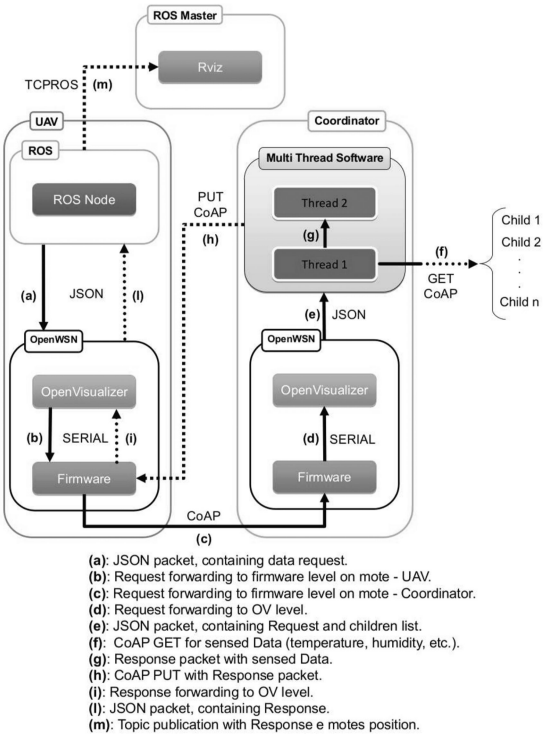

Fig. 3 *Overall data flow of the IoT-aided robotic system under test*
(a) JSON packet, containing data request, ***(b)*** Request forwarding to firmware level on mote – UAV, ***(c)*** Request forwarding to firmware level on mote – coordinator, ***(d)*** Request forwarding to OpenVisualizer (OV) level, ***(e)*** JSON packet, containing request and children list, ***(f)*** CoAP GET for sensed data (temperature, humidity etc.), ***(g)*** Response packet with sensed data, ***(h)*** CoAP PUT with response packet, ***(i)*** Response forwarding to OV level, ***(l)*** JSON packet, containing response, ***(m)*** Topic publication with response e motes position

sensors. Moreover, the OpenWSN protocol stack is able to run on top of different OSs, i.e. openon and FreeRTOS. While the latter integrates real-time functionalities and it is designed to run on more capable devices (i.e. Zolertia Z1 [http://zolertia.sourceforge.net/wiki/images/e/e8/Z1_RevC_Datasheet.pdf] or OpenMote-CC2538 [http://www.openmote.com/hardware/openmote-cc2538-en.html]), the former has been chosen as it is more suitable for constrained hardware platform such as TelosB.

OpenWSN's firmware has been modified to make each node able to execute specific actions. Some of the changes are common to all nodes (e.g. weakening TX transmission power), others not. In details, child nodes are allowed to detect sensor values. To this aim, the serial peripheral interface (SPI) is used to perform communications with the MCU, enabling master/slave interaction. For the sake of completeness, the MCU is the master and controls both the bus and its clock, enabling start and stop interrupt to communication. The slave, indeed, answers to commands received based on the timing imposed by the MCU. A specific application has been created which is able to read the data and answer to the performed CoAP GET. The firmware of the UAV node, instead, has been modified allowing commands interception through serial interface and receiving CoAP PUT data from the coordinator.

The extremely rich hardware and software setup deserves a precise orchestration to work properly. In general, to make a node belonging to the network out of the UAV, the onboard computer has to be equipped with the IoT device (for instance, TelosB), connected via USB port. It is essential to create an ROS node able to interact with OpenVisualizer. Using ROS implies the possibility of sending requests to the network coordinator that answers with sensors data from the nodes in the IoT network. In the peer-to-peer communication scheme of ROS, the main actors are the ROS master and the nodes, performing specific tasks according to the publish–subscribe communication paradigm. ROS master registers nodes and allows messages exchange using publish–subscribe schema; for instance, NODE A publishes a message on the TOPIC '*n*' and this message is received by NODE B that subscribed to the same topic. In details, when a node is connected to ROS, it publishes messages on a topic, whereas all other affected nodes perform a subscription. After that, data are received. In this scenario, multiple publishers and subscribers are allowed and they are not necessarily aware of each other. Node connections are provided via direct links, whereas the master is only in charge of searching information, in a similar way to a domain name system server. In fact, it uses name resolution and IPv6 addresses through a database. The nodes that subscribe to a topic, asking for a connection to the publisher, use a dedicated transport layer solution, a protocol called transfer control protocol robot operating system (TCPROS) which involves a standard transfer control protocol/Internet protocol socket and is specifically meant for ROS messages and services.

The software suite is reported in Table 2.

3.3 Procedures

Fig. 3 describes the data flow within the system events. To serve different kinds of missions, three types of requests can be issued by the UAV after it joins the IoT network, i.e.:

- Discovery Request: the coordinator provides information about the nodes and their resources.
- Data Request: coordinator sends the value of a sensor on a specific node.
- All Data Request: the response packet contains all sensor values coming from all child nodes.

Going through Fig. 3, it can be seen that the UAV node exchanges information with OpenVisualizer (a) using JavaScript Object Notation (JSON)-format files, which has been chosen to guarantee ease of use and wide compatibility. The content of the request type is:

- Discovery Request (#R#).
- All Data Request (#A#).
- Data Request (#idmote:id_ res#).

As data are received by the OpenWSN software layer, they are forwarded via serial interface to a specific application (b), running on top of the physical mote, which is in charge of processing the packet containing the request and sending it to the coordinator (c). The packet is forwarded (d) from the firmware to the OpenWSN software level, where it is enriched of the motes connected list and leveraging JSON packet is sent (e) to a specific multi-threaded software in charge of both processing the request and query child nodes. The former gets values from the peripheral nodes using a CoAP GET operation (f) and forwards them to the second thread (g). The latter sends requested data performing a PUT CoAP (h). The packet sent contains information about the id of the mote sending data together with sensed values. The response is forwarded to the ROS node via paths (i) and (l) which publishes messages on a topic for the ROS master. Fig. 4 represents the flowchart illustrating the functions used by the ROS node and the network coordinator while communicating.

In details, the software procedure starts with the UAV node, sending a JSON request. The discovery features, added to OpenVisualizer, receive the request and processes it. As the request is sent to the mote, the firmware level gets involved in order to forward it through a specific method aimed at sending requests. In particular, it is working for both receiving data and building the packet containing the request. The network coordinator is, so, reached via CoAP. It processes the message, also retrieving the

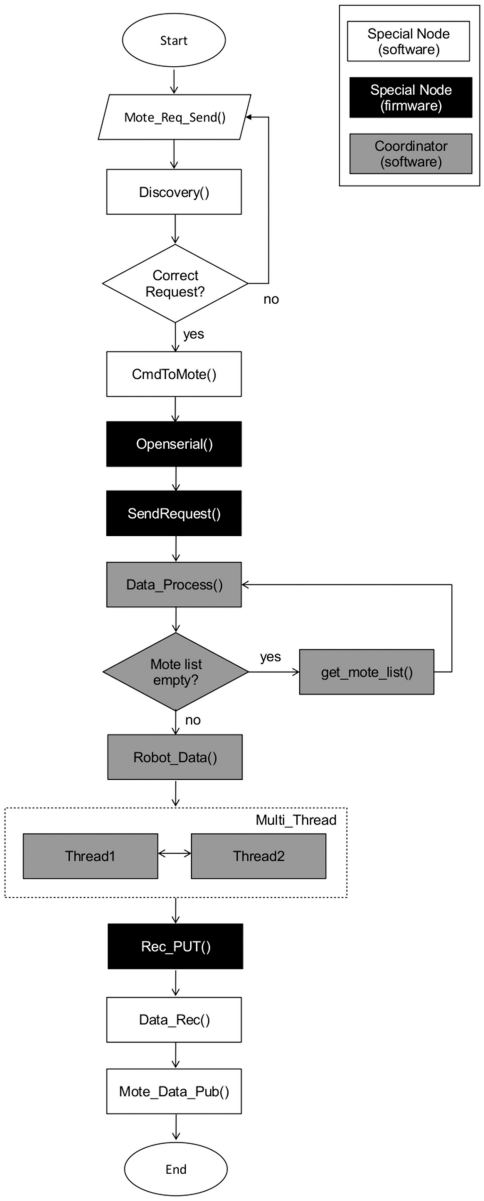

Fig. 4 *Communication procedures between the special node and the network coordinator – flowchart*

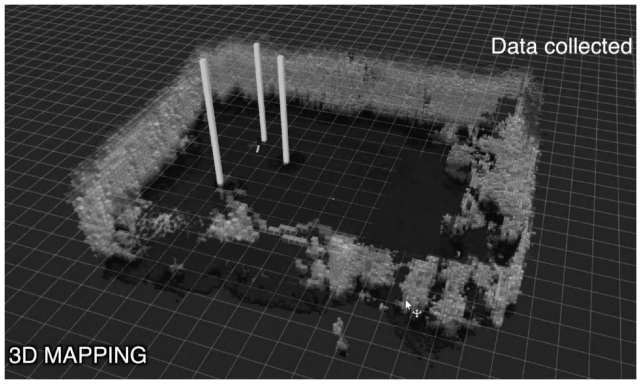

Fig. 5 *3D mapping of the surveyed area. Pillars identify collected data points*

addresses of child nodes connected. A packet containing the list of the motes and the request from the ROS node is created and forwarded. Before leaving the network coordinator, the software layer operates in a multi-threaded fashion. In fact, it is divided into two different parts, one requiring, via GET CoAP, sensor values to peripheral nodes and saving them in a shared library, the other reading data from the library, creating a packet and sending it, via PUT CoAP, to the mote connected to the UAV. Data flow proceeds on the onboard IoT device, as a dedicated function receives the packet and sends it to OpenVisualizer. Here, data are processed and converted into JSON format. The UAV node, waiting for the information to be communicated, is able to receive them. According to the specific request sent, it receives the packet and communicates the values to ROS master (m). At this point of the data flow, the software solution recreates the mapped area thus allowing the visualisation of what has been acquired. The outcome is shown in Fig. 5.

4 Experimental performance evaluation

The testbed described in Section 3 has been experimentally evaluated. To this end, a mission has been defined for the UAV and the following key performance indices have been analysed during the mission: (i) memory footprint, (ii) IoT-aided system power consumption, (iii) network joining time, (iv) QoS and (v) system reliability.

For the sake of clarity, end nodes in the network are identified with the last four digits of the IPv6 address. For instance:

- Coordinator: 5201.
- UAV: 1C01.
- Child nodes: 4F01 (Child1), 0001 (Child2) and 9D01 (Child3).

4.1 Memory footprint

IoT devices are often constrained in terms of available memory. In particular, TelosB is equipped with 10 kB RAM and 48 kB read-only memory only. For this reason, code optimisation is mandatory. The footprints of compiled firmware versions running on motes resulted to be fully compatible with the chosen hardware platform. Fig. 6 reports the amount of memory used on the different nodes of our system and demonstrates that it fits TelosB features.

It should be pointed out that TelosB can be considered as a worst case as regards memory availability, since it is a very old platform. Therefore, the use of more powerful IoT devices (i.e. Zolertia Z1, OpenMote) would be straightforward based on these results.

4.2 IoT-aided system power consumption

The activity of the onboard IoT device was monitored, while it was fed through the USB interface, in order to measure the drawn current. The power module, introduced in Section 3, allows monitoring battery voltage and current consumption. Calculating the difference between the value of the current consumed with and without TelosB, it is possible to evaluate its absorption. The maximum and minimum values measured, in the absence of the device, are 0.31 and 0.29 A, respectively (see Fig. 7*a*). The relief refers to a total time of 100 s and the average value taken of 0.302 A. The same procedure was applied after connecting TelosB obtaining similar values (as shown in Fig. 7*b*).

Monitoring the current absorption by the IoT device onboard of the UAV allows to verify to what extent the contribution of the IoT device can impair the lifetime of the UAV batteries. With reference to this issues, it is worth to point out that the additional energy spent by the UAV to carry the IoT device has been neglected in our analysis. In fact, TelosB's weight is as low as 23 g, and its physical dimensions are $65 \times 31 \times 6$ mm^3. In other words, only the current drained by the IoT node onboard of the UAV has been considered in this experimental campaign. Obtained results demonstrate that adding the mote to the UAV does not significantly inflate the overall current consumption, as it is as low as 3 mA, and causes a negligible overhead of 3.3%.

4.3 Network joining time

Network joining time is important for the complete characterisation of the IoT network. Network joining procedure involves several components, from MAC-layer synchronisation to RPL upward and downward paths formation. In particular, after the joining phase, nodes are synchronised and the network is fully operative. To

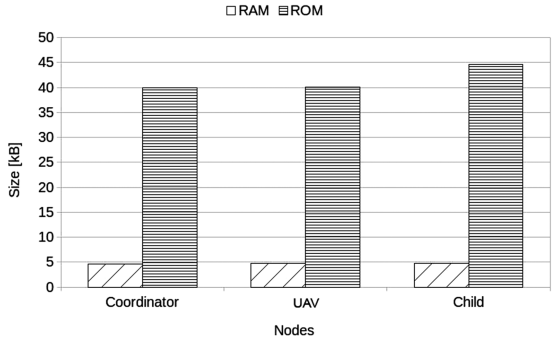

Fig. 6 *TelosB memory footprint*

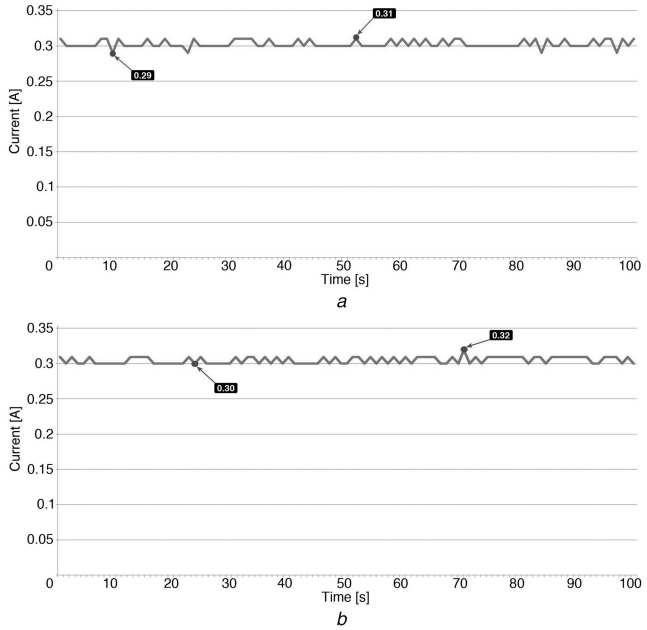

Fig. 7 *Current consumption of onboard devices, excluding motors*
(a) Without TelosB,
(b) With TelosB

Table 3 Network joining time (measured in seconds)

	UAV 1C01	Child1 4F01	Child2 0001	Child3 9D01
1	37.65	36.65	43.88	37.4
2	35.09	74.1	42.7	36.9
3	45.72	35.18	55.49	75
4	30.95	56.51	40.48	29.3
5	22.6	15.5	15.5	18.8
6	56	61	56	29.3
7	23.5	22.7	27.9	39.8
8	31.6	27.5	27.5	11.4
9	15.9	15.9	15.9	53.1
10	27.6	35.7	30.8	11.6

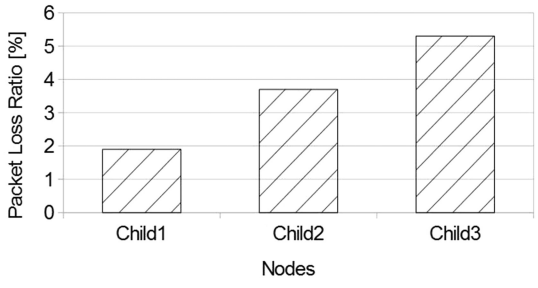

Fig. 8 *Lost packets percentage for each child|MAC layer*

estimate this figure of merit, for each node it has calculated the period of time between the time instant at which the first packet has been sent and the reception on the root node. Table 3 reports measured values. In particular, network joining times range from a minimum of 11 s to a maximum of 74. Afterwards, the IoT network is perfectly set up, in terms of shared hopping sequence and schedule; moreover, RPL paths are established. Then, the network begins its monitoring activity on the area of interest.

It is worth noting that synchronisation happens only once, at the very beginning of the network activity. Moreover, measured values are fully compatible with the maximum flight range for the chosen UAV. Indeed, even if the IoT-aided robotics system flew within the area of interest before the network is set up, it would be able to complete the gathering data mission anyway. These experimental evidences can be considered a preliminary assessment to the deployment of much more complex networks covering larger areas. Finally, it is worth to remark that advanced network formation schema can be adopted (as in [41]) to reduce network joining time in 6TiSCH networks.

4.4 Quality of service

To experimentally evaluate the QoS of the testbed, it was crucial to test communication tasks executed at the application layer. To execute this evaluation under strenuous conditions, MAC-layer retransmissions were disabled. In particular, two operations belonging to the CoAP protocol have been considered: GET and PUT. For each of them, the PLR contributed by the protocol stack and execution time were measured.

To detect the execution time of the GET operation, a special application was created on the network coordinator. Each child node is running a corresponding application able to detect the sensor data and to respond to the GET. Fig. 8 reports the PLR for each child node and shows that in the worst case <6% of packets were lost: this achievement is in line with the requirements of environmental monitoring systems and can be further improved (if necessary), thanks to automatic retransmission request mechanisms.

The PUT operation is triggered by the multi-thread software that sends to the UAV the packet containing sensor data coming from network nodes (as shown in Fig. 3). To this end, a dedicated application monitors the effective reception for every request sent by the UAV and triggers retransmissions, if needed. In this way, a 100% reliability is pursued at the application layer. To be more specific, three different packet types have been defined to map the different requests by the UAV: discovery, with the list of all child nodes with their resources, single data, containing a resource belonging to a child node and all data, containing all the values of all sensor nodes.

Fig. 9*a* shows the PLR contributed by the protocol stack in UAV to network data exchange: here, the PLR is higher than in the previous case (i.e. child to coordinator communications) because the UAV is mobile and can be farther from the coordinator than a static child node. In any case, this pretty high PLR is completely compensated, thanks to the adoption of application layer packet retransmissions. Finally, it is worth remarking that the resulting execution time reported in Fig. 9*b* is always <3.5 s, which is compliant to many industrial use cases [3].

4.5 System reliability

The system reliability test phase has been conceived for testing the whole data flow, thus verifying that the system is properly configured. In this phase, the IoT-aided robotics unit performs every task in the envisioned mission plan. To this aim, a dedicated routine implementing the aforedescribed procedures (see Section 3) has been created. It is specifically aimed at measuring the average execution time and the PLR. Functionally speaking, the ROS node initiates the procedure sending a request to the node via the network coordinator and waits for the answer. When data are received, the packet is processed and sent to the ROS visualizer node, which is in charge of publishing the results. In this phase, no packets have been lost and the calculated average execution time is 35.9 s, regardless of data type.

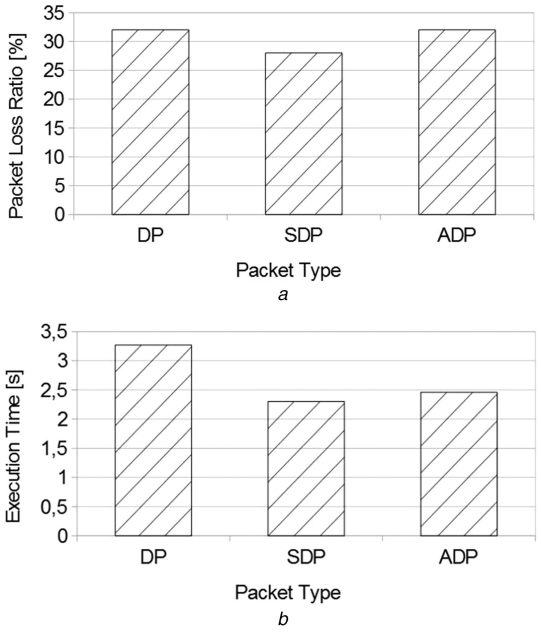

Fig. 9 *PLR contributed by the protocol stack in UAV to network data exchange*

(a) PLR, ***(b)*** Execution time in PUT test for discovery packet (DP), single data packet (SDP) and all data packet (ADP)

The tests carried out prove effective data collection and processing and the IoT-aided robotics solution resulted correctly configured. Moreover, experimental evidences prove that the mission lasts considerably less than the average flight time of the UAV. In fact, with the supplied battery, the maximum flight range is 8 min. As a consequence, it is possible to start daring larger-scale deployments in terms of both are as of interest and number of IoT devices and the number of UAVs.

5 Conclusions and future research

This paper paves on the strong motivation of testing IoT solutions in operating scenarios in which UAVs and IoT devices interact with the environment they are within, gathering data coming from sensors. The proposed case study has been investigated through several hardware instruments including an UAV and IoT devices. They have been used in conjunction with suitable software tools, namely ROS and OpenWSN, to realise a complete open source interface able to optimise communication schemes. The intermediate software layer, running on top of the network coordinator, can directly interact with devices using CoAP protocol.

The outcomes of the experimental campaign are notable. It has been proven that patching the robotic system with a mote has no harmful contribution and the resulting IoT-aided system is stable and fully working. The deploy of the IoT network has been performed in a short period of time, which represents a preliminary measure for a larger network development. Moreover, it has been proven that surveying and patrolling activities can be performed with good performance in terms of data gathering and messages delivery, compatible with those of the single entities operating in stand-alone situations.

Nevertheless, several enhancements to the described set up can be considered for future work. First of all, the proposed solution has to be tested in larger, outdoor contexts, as this will prove scalability to be a reachable target. Moreover, it might be useful to consider different solutions in order to guarantee longer mission durations including advanced network formation schema (as those proposed in [41]) to reduce joining time to 6TiSCH networks. One of the main research prospective is to coordinate multiple UAVs flying together at the same time and gathering data from a wider and more populated network. With a specific reference to IoT solutions, the hardware platform used here has very limited computational capabilities, when compared with others. For this reason, one of the possibilities is to test higher profile boards.

6 Acknowledgments

The financial support of the FP7 ERA-NET ICT-AGRI 2 through the grant Simultaneous Safety and Surveying for Collaborative Agricultural Vehicles (Id. 29839) (S3-CAV) is gratefully acknowledged. This work was also partially supported by the BONVOYAGE project, which received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement 63586.

7 References

[1] Atzori, L., Iera, A., Morabito, G.: 'The Internet of things: a survey', *Comput. Netw.*, 2010, **54**, pp. 2787–2805

[2] Shue, S., Conrad, J.M.: 'A survey of robotic applications in wireless sensor networks'. 2013 Proc. of IEEE Southeastcon, April 2013

[3] Palattella, M., Accettura, N., Vilajosana, X., *et al.*: 'Standardized protocol stack for the Internet of (important) things', *IEEE Commun. Surv. Tutor.*, 2013, **15**, (3), pp. 1389–1406

[4] Nukala, R., Panduru, K., Shields, A., *et al.*: 'Internet of things: a review from 'farm to fork'. 2016 27th Irish Signals and Systems Conf. (ISSC), June 2016, pp. 1–6

[5] Watteyne, T., Palattella, M.R., Grieco, L.A.: 'Using IEEE 802.15.4e time-slotted channel hopping (TSCH) in the Internet of things (IoT): problem statement', Internet Engineering Task Force (IETF) – Request for Comments: 7554, 2015

[6] Palattella, M.R., Accettura, N., Grieco, L.A., *et al.*: 'On optimal scheduling in duty-cycled industrial IoT applications using IEEE802.15.4e TSCH', *IEEE Sens. J.*, 2013, **13**, (10), pp. 3655–3666

[7] Wu, F.Y., Asada, H.H.: 'Implicit and intuitive grasp posture control for wearable robotic fingers: a data-driven method using partial least squares', *IEEE Trans. Robot.*, 2016, **32**, (1), pp. 176–186

[8] Fard, M.J., Ameri, S., Chinnam, R.B., *et al.*: 'Soft boundary approach for unsupervised gesture segmentation in robotic-assisted surgery', *IEEE Robot. Autom. Lett.*, 2017, **2**, (1), pp. 171–178

[9] Macwan, A., Vilela, J., Nejat, G., *et al.*: 'A multirobot path-planning strategy for autonomous wilderness search and rescue', *IEEE Trans. Cybern.*, 2015, **45**, (9), pp. 1784–1797

[10] Petitti, A., Franchi, A., Di Paola, D., *et al.*: 'Decentralized motion control for cooperative manipulation with a team of networked mobile manipulators'. 2014 IEEE Int. Conf. on Robotics and Automation (ICRA), May 2016, pp. 441–446

[11] Shneier, M.O., Bostelman, R.V.: 'Literature review of mobile robots for manufacturing'. Technical Report, NISTIR – 8022, NIST Pubs, 2015

[12] Reina, G., Milella, A., Rouveure, R., *et al.*: 'Ambient awareness for agricultural robotic vehicles', *Biosyst. Eng.*, 2016, **146**, pp. 114–132, special Issue: Advances in Robotic Agriculture for Crops. Available at http://www.sciencedirect.com/science/article/pii/S1537511015001889, DOI: 10.1016/j.biosystemseng.2015.12.010, accessed 4 May 2017

[13] Dunbabin, M., Marques, L.: 'Robots for environmental monitoring: significant advancements and applications', *IEEE Robot. Autom. Mag.*, 2012, **19**, (1), pp. 24–39

[14] Greenblatt, N.A.: 'Self-driving cars and the law', *IEEE Spectr.*, 2016, **53**, (2), pp. 46–51

[15] D'Andrea, R.: 'Mobile-robot-enabled smart warehouses', in Samad, T., Annaswamy, A., (EDs.): *'The impact of control technology,* (2014)

[16] Kohl, A.M., Pettersen, K.Y., Kelasidi, E., *et al.*: 'Planar path following of underwater snake robots in the presence of ocean currents', *IEEE Robot. Autom. Lett.*, 2016, **1**, (1), pp. 383–390

[17] Fernández, D.C., Hollinger, G.A.: 'Model predictive control for underwater robots in ocean waves', *IEEE Robot. Autom. Lett.*, 2017, **2**, (1), pp. 88–95

[18] Luo, J., Yao, J., Peng, Y., *et al.*: 'Modeling of an anti-interference spherical robot for polar region scientific research'. 2014 IEEE Int. Conf. on Information and Automation (ICIA), July 2014, pp. 1300–1305

[19] Shah, J.A., Saleh, J.H., Hoffman, J.A.: 'Review and synthesis of considerations in architecting heterogeneous teams of humans and robots for optimal space exploration', *IEEE Trans. Syst. Man Cybern. C (Appl. Rev.)*, 2007, **37**, (5), pp. 779–793

[20] Rentschler, M.E., Platt, S.R., Berg, K., *et al.*: 'Miniature in vivo robots for remote and harsh environments', *IEEE Trans. Inf. Technol. Biomed.*, 2008, **12**, (1), pp. 66–75

[21] Bullo, F., Cortés, J., Martínez, S.: '*Distributed control of robotic networks, ser. applied mathematics series*' (Princeton University Press, 2009). Available at http://coordinationbook.info

[22] Grieco, L.A., Rizzo, A., Colucci, S., *et al.*: 'Iot-aided robotics applications: technological implications, target domains and open issues', *Comput. Commun.*, 2014, **54**, pp. 32–47

[23] Tahar, K.N., Ahmad, A., Akib, W.A.A.W.M., *et al.*: 'Aerial mapping using autonomous fixed-wing unmanned aerial vehicle'. 2012 IEEE Eighth Int. Colloquium on Signal Processing and its Applications, March 2012, pp. 164–168

[24] Watteyne, T., Weiss, J., Doherty, L., *et al.*: 'Industrial IEEE802.15.4e networks: performance and trade-offs'. IEEE Int. Conf. on Communications (IEEE ICC), vol. Internet of Things Symp., London, UK, June 2015, pp. 8–12

[25] 'IPv6 over low power WPAN (6Lo)', IETF 6Lo working group, 2016. Available at http://datatracker.ietf.org/wg/6lo/, accessed 4 May 2017
[26] 'Routing over low power and lossy networks (roll)', IETF ROLL working group, 2016. Available at http://datatracker.ietf.org/wg/roll/, accessed 4 May 2017
[27] 'Constrained RESTful environments (CoRE)', IETF CoRE working group, 2016. Available at http://datatracker.ietf.org/wg/core/, accessed 4 May 2017
[28] Jayakumar, H., Raha, A., Kim, Y., *et al.*: 'Energy-efficient system design for IoT devices'. Proc. IEEE Asia and South Pacific Design Automation Conf. (ASP-DAC), January 2016, pp. 298–301
[29] Palattella, M.R., Watteyne, T., Wang, Q., *et al.*: 'On-the-fly bandwidth reservation for 6TISCH wireless industrial networks', *IEEE Sens. J.*, 2016, **16**, (2), pp. 550–560
[30] Shelby, Z., Hartke, K., Bormann, C., *et al.*: 'Constrained application protocol (CoAP)'. IETF CoRE Working Group, February 2011
[31] Postel, J.: 'User datagram protocol', Internet engineering task force – request for comments: 768, 1980
[32] 'IEEE Standard for Local and Metropolitan Area Networks – Part 15.4: 'Low-rate wireless personal area networks (LR-WPANs)', Amendment 1: MAC Sublayer', 16 April 2012
[33] Kehoe, B., Patil, S., Abbeel, P., *et al.*: 'A survey of research on cloud robotics and automation', *IEEE Trans. Autom. Sci. Eng.*, 2015, **12**, (2), pp. 398–409
[34] Ray, P.P.: 'Internet of robotic things: concept, technologies, and challenges', *IEEE Access*, 2016, **4**, pp. 9489–9500
[35] Razafimandimby, C., Loscri, V., Vegni, A.M.: 'A neural network and IoT based scheme for performance assessment in Internet of robotic things'. 2016 IEEE First Int. Conf. on Internet-of-Things Design and Implementation (IoTDI), April 2016, pp. 241–246
[36] Santoso, F.: 'Range-only distributed navigation protocol for uniform coverage in wireless sensor networks', *IET Wirel. Sens. Syst.*, 2015, **5**, (1), pp. 20–30
[37] Chung, J.M., Nam, Y., Park, K., *et al.*: 'Exploration time reduction and sustainability enhancement of cooperative clustered multiple robot sensor networks', *IEEE Netw.*, 2012, **26**, (3), pp. 41–48
[38] Li, X., Lille, I., Falcon, R., *et al.*: 'Servicing wireless sensor networks by mobile robots', *IEEE Commun. Mag.*, 2012, **50**, (7), pp. 147–154
[39] Mathur, P., Nielsen, R.H., Prasad, N.R., *et al.*: 'Data collection using miniature aerial vehicles in wireless sensor networks', *IET Wirel. Sens. Syst.*, 2016, **6**, (1), pp. 17–25
[40] Watteyne, T., Vilajosana, X., Kerkez, B., *et al.*: 'OpenWSN: a standards-based low-power wireless development environment', *Wiley Trans. Emerging Telecommun. Technol.*, 2012, **23**, (5), pp. 480–493
[41] Vogli, E., Ribezzo, G., Grieco, L.A., *et al.*: 'Fast join and synchronization schema in the IEEE 802.15.4e MAC'. Proc. IEEE Wireless Communications and Networking Conf., WCNC, Workshop on Energy Efficiency in the Internet of Things, and Internet of Things for Energy Efficiency, E2IoT, New Orleans, LA, USA, March 2015