# A Hardware-Software WSN Platform for Machine and Structural Monitoring

David Rojas* and John Barrett†
Nimbus Centre, Cork Institute of Technology, Ireland
Email: *f.rojas-calvente@mycit.ie, †john.barrett@cit.ie

***Abstract*—Traditional wired vibration and acoustic sensors used for machine and structural monitoring are currently being replaced by low-cost MEMS-based wireless sensor networks (WSN). However, existing platforms are lacking in computing capabilities and integration, as well as the necessary software features to manage wireless sensing experiments. In this paper, we present a novel hardware-software platform designed to monitor machinery in remote deployments and expedite collection of experimental data, which could also be used for structural monitoring. The hardware module is composed of a single PCB with an IEEE 802.15.4-compatible microcontroller, waterproof Micro-USB connector, battery, battery charger/monitor, humidity/temperature sensor, IMU, and a MEMS microphone. The software developed allows for wireless experiment control and data collection through a gateway node connected to a laptop. Additionally, the user interface supports the placement of the nodes in a 3D view of the environment, as well as visualisation of the collected data. The platform was tested in the laboratory in two different motor setups by measuring vibration and sound in normal operation, showing that the system can facilitate the execution of sensing experiments in rotating machinery and similar equipment.**

## I. Introduction

Vibration and acoustic sensors have been widely used for prognostics and fault diagnosis of bearings [2], motors [3], and different types of machinery [4], [5]. To reduce cost and ease installation, high-end wired sensor systems for vibration based machine and structural monitoring are being replaced with low-cost wireless sensor networks (WSN) using MEMS and other COTS sensors [6]–[8]. This becomes important specially in remote locations and off-shore deployments such as marine platforms and ships, where their difficult access and harsh environment conditions further complicate their maintenance and monitoring.

In [9], a WSN is used to monitor the condition of a marine gearbox, with nodes based on the assembly of several off-the-shelf boards: a Zigbee module, an accelerometer, and an energy harvester. Similarly, Severino et al. [10] showed a platform for structural health monitoring using a TelosB [11] mote connected to an acquisition board and accelerometer. While these experiments demonstrate the feasibility of using wireless sensors and MEMS to monitor these environments, most of the platforms used present a series of drawbacks: they are either based on old platforms developed for networking research with limited processing capabilities and memory (e.g. TelosB and MICAz), which limits also the sampling rate of the sensors, or on more powerful platforms with higher power consumption and that do not support current standard WSN operating systems such as Contiki-OS [12]. Moreover, accelerometer and acoustic sensors are rarely used together in these studies, even though the literature shows methods that can combine both sources of data to improve fault detection [4], [13], [14]. In addition, the use of different modules for communications, acquisition, and sensors attached with cables or connectors instead of a single integrated board complicates the packaging and encapsulation, as well as increasing the cost and power consumption.

To accommodate the high data rate of the sensors used for machine and structural monitoring, a hardware-software co-design is necessary that ensures the complete system can perform as needed while keeping a low power consumption and compatibility with the standards. Although some work has been done in developing hardware-software platforms for WSN [15], [16], most of the software tools are designed to work for the mentioned obsolete platforms and they are focused on networking, protocols, and wireless planning and characterisation instead of sensing.

In this paper, we developed a hardware and software architecture designed primarily for monitoring marine energy platforms and their machinery, which can also be used for more general monitoring of ships, industrial machinery, and structural health. The hardware module is a single board based on a state-of-the-art IEEE 802.15.4-compatible microcontroller with MEMS vibration, microphone, and humidity/temperature sensors, battery, and battery management, encased in a custom 3D printed box designed for easy encapsulation. The firmware and software developed for this platform allow for wireless data collection and visualisation, experiment control, and 3D view support of the environment and node location.

The remainder of the paper is structured as follows: Section II presents the overall hardware-software system characteristics. Section III describes the custom wireless node designed and sensors used, while Section IV introduces the main features of the firmware/software platform. In Section V the system is evaluated in two laboratory setups, with a motor attached to a wind turbine and a gearbox. Section VI ends with the conclusions and improvement plans for the platform.

This material is based upon works supported by the Science Foundation Ireland under Grant No. 12/RC/2302, the Marine Renewable Energy Ireland (MaREI) Centre research programme [1].

## II. Overall system design

The system is composed of a sensing node ready to be deployed in harsh environments, which could sense short bursts of vibration and audio from the machinery or structure, store the data, and send them wirelessly to the gateway on demand, to be analysed later to detect and predict failures.

The hardware module is based on a single PCB for CPU, communications, sensors, and power management, contained along with the battery in an ABS plastic box with exposed antenna and charging connector. This simplifies the assembly, lowers production costs, and facilities encapsulation to protect it from humidity and dust.

The software supporting the hardware module will have a firmware layer running in the microcontroller, based on Contiki-OS, as well as a script and user interface that will handle the requests from a PC to the nodes. This firmware/software architecture will have two main modes of operation:

- On-demand experiments: The PC connected to a gateway node will send commands to a specific node to start a sensing experiment. After the experiment is finished, the gateway will download the data.
- Periodic unattended experiments: The node will receive from the PC gateway the experiment parameters, and after that the node will repeatedly execute the command with the specified period. The list of commands will be stored in the node memory, thus the experiments can start autonomously if the node is reset or turned off/on.

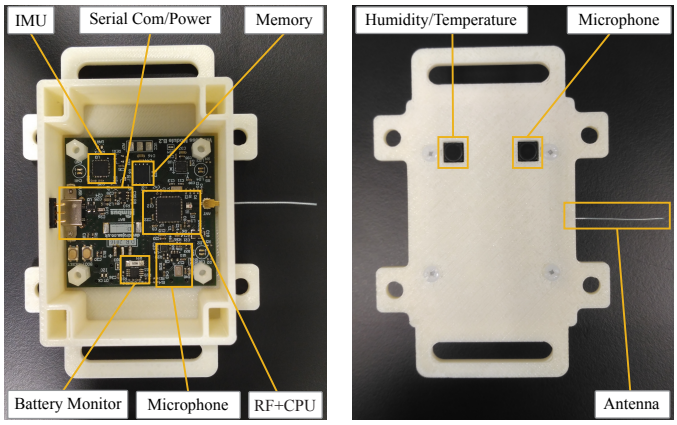
All the communications from the gateway to the nodes will be done wirelessly and will support different commands to facilitate the control and configuration of the nodes, such as recording audio and vibration data, sensor calibration, logging battery level, setting transmission power, etc.

## III. Hardware platform

The electronics are assembled on a 50 x 50 mm four layer PCB, with most of the components on its top layer, as it can be seen in Fig. 1a. The bottom layer holds only the humidity/temperature sensor. Although the microphone itself is placed on the top layer, to capture the sound there is a through hole to the bottom layer, next to the humidity/temperature sensor. Both sensor openings are protected by a cap from dust and water, as shown in Fig. 1b. The external antenna is connected to an UF.L connector, and in this way it can be located at a distance or replaced by a higher gain one if necessary. The firmware uploading and battery charging is done through a waterproof microUSB connector, to allow to fully seal the module. The details of the assembly are explained in Section III-D.

### A. CPU, communications, and memory

The processing unit used is the CC2538, a low-power ARM Cortex-M3 MCU with 32 KB RAM and 512 KB Flash, which includes a 2.4 GHz IEEE 802.15.4 radio and a wide range of peripherals, including an $I^2C$ and SPI bus to interface with digital sensors, and a 7.5 KHz ADC necessary to sample




(a) (b)

Fig. 1. (a) Top PCB inside the box (b) Bottom with covered sensors

the microphone audio. This MCU was chosen mainly for its increased processing power over old MSP430-based platforms, its integrated radio that reduces cost and power consumption, and its good Contiki-OS support.

A low power AT45DB641E flash memory is also added, connected to the SPI bus, to be able to store the data of the experiments as well as configuration parameters. This memory has a size of 64 Mbit, which allows to store up to 90 samples of 5 s bursts of combined accelerometer and audio data. In addition to the main memory, it also contains two SRAM buffers of 256 bytes each, which can increase the system ability to write a continuous data stream.

### B. Power management

To power the module, a 4.2V rechargeable lithium polymer battery of 4500 mAh capacity is used, with its voltage regulated by an MCP1700 LDO regulator. The battery charging is handled by an MCP73831 charger IC, with two SMD LEDs to indicate the battery charging state (charging/charged). Additionally, a DS2745 battery monitor connected to the $I^2C$ bus provides the battery voltage, as well as the instant and accumulated current consumed by the node, which is useful to predict its battery life.

The module power consumption is approximately 25 mA when sampling data, 2 mA in command listening mode, and 35 mA when transmitting data. This allows for around 90 days of operation if configured to capture and save samples of 5 s of audio and accelerometer at a rate of 1 sample/day, which corresponds also with the maximum number of samples the memory can store.

### C. Sensors

*1) Microphone:* MEMS microphones are commonly designed for smartphones, headsets, and similar applications, therefore they are optimised for human voice and environments with a medium level volume. Because our application targets heavy machinery, a digital MEMS microphone can saturate its output when sensing in very noisy environments or placed on loud machines. Thus, we need to use an analog MEMS microphone that allows us to design the conditioning

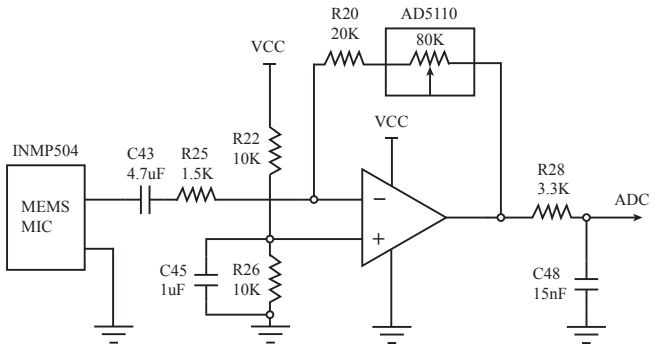



Fig. 2. Microphone conditioning circuit with configurable gain.

circuit with the needed gain. This gain though will be dependent on the specific environment and conditions, and to maximise signal excursion and resolution without saturating the signal the recommended amplifying circuit will need to be modified to allow for a variable gain.

The selected microphone is the INMP504, connected to the 12 bits ADC of the microcontroller through the signal conditioning circuit shown in Fig. 2, formed by an opamp-based inverter amplifier with a high-pass filter at the input and a low-pass filter at the output. The gain of the amplifier is adjusted by an AD5110 $I^2C$ digital potentiometer controlled by the microcontroller, connected in series to the resistor R22. This yields a maximum gain of 58.8 and a minimum of 12. The minimum gain value was designed to avoid saturation if the captured sound is near the maximum value supported by the microphone, while the maximum gain corresponds to the one fixed in the evaluation board of the microphone for standard use. The C43-R25 high-pass filter removes the DC before the resistor divider R22-R26 sets it to VCC/2 for maximum signal excursion. As the ADC of the CC2538 has a maximum sampling rate of 7.5 KHz, we need an anti-aliasing filter before the signal is digitised. This is done by the R28-C48 filter, resulting in a 3.1 KHz microphone bandwidth, which is sufficient to cover noise in machinery as its sound is usually low in frequency.

To adjust the gain, an auto-calibration mechanism is developed in the firmware. Before the first experiment in a new environment or conditions, the microphone will sample 5 s of audio with the gain set to the minimum value, measure the maximum signal received and calculate with this the corresponding gain for this signal to have the maximum excursion. After this the value of the potentiometer will be saved in its own EEPROM and will keep its current value until the calibration command is launched again.

*2) IMU:* As shown in the literature [2], [3], [9], a bandwidth of 100-200 Hz is sufficient to detect faults using vibration data in motors, bearings, and similar machinery, while the requirement for structural health monitoring is even lower [7], [10]. Hence, a MEMS accelerometer provides sufficient bandwidth for both machine and structural vibration. We selected the LSM9DS1 connected to the $I^2C$ bus, a full MEMS IMU that also contains a gyroscope and magnetometer instead of a simple accelerometer, to be able to use the module also for monitoring the position and sway of offshore platforms and floating devices. Although this IC allows for an accelerometer sampling rate up to 952 Hz, due to the speed limitations of the microcontroller and memory the sensor is configured to be used at 476 Hz, which is the maximum rate that can be used without losing frames, yielding a sensing bandwidth of 211 Hz. The accelerometer can be used up to $\pm 16$ g, but it is configured by default at a full scale of $\pm 8$ g, which is enough for machine vibration monitoring.

*3) Temperature and humidity:* The sensor selected for humidity and temperature monitoring is the SHT21, as the next generation of the SHT11 found in older WSN platforms such as the TelosB. This sensor has a 0.01 °C resolution for temperature and a 0.04% for relative humidity. To protect it against water, dust, and other contaminants, the sensor has a companion protective cap that can be easily attached to the PCB, consisting of a single piece of PBT thermoplastic and a filter membrane.

### D. Module assembly

Since the node is designed to operate in harsh environments, a custom ABS 3D printed box is designed to fit the electronics and battery inside, with only the microUSB connector, on/off switch, antenna, humidity/temperature, and microphone opening exposed. The humidity/temperature and microphone holes however are covered by two protective caps, as it was shown in Fig. 1b. As there is no standard way to protect MEMS microphones, a simple audio test covering it with the same cap used for the humidity/temperature sensor proved that it did not have any noticeable effect in the recorded sound level or frequency response compared with the opening without cover.

Fig. 3 shows an exploded view of the whole assembly. The PCB sits flat at the bottom of the enclosure, attached with four plastic screws and spacers. The battery is placed on top, held together between the spacers, the top enclosure, and another four screws that close the box going from the lid to the PCB spacers. To be able to see the LED indicators when the box is closed, three light guides inserted into holes in the lid interface with the SMD LEDs in the PCB. Four flaps with holes extend from two sides of the box, as well as two larger handles on the other sides, which can be used to attach de node to the machine or structure with cable ties or screws. Finally, to facilitate the encapsulation of the module for extra protection against humidity and salt, two slits cut out on the top enclosure enable the pouring of epoxy inside the box after the module is fully assembled.

## IV. Firmware and software platform

### A. Firmware

The firmware running on the CC2538 microcontroller is built on top of Contiki-OS, an open source operating system for the Internet of Things, optimised for resource-constrained

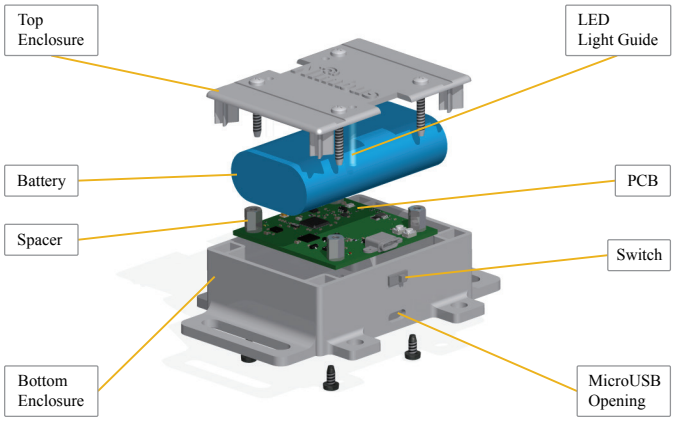



Fig. 3. Sensor module assembly.

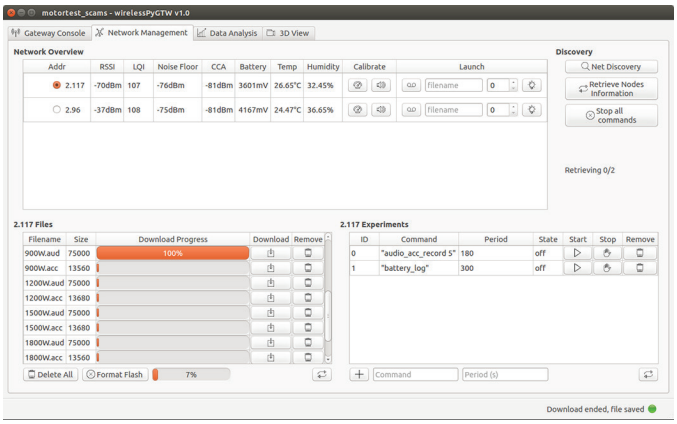



Fig. 4. UI Network view.

devices and supporting most common low-power communication standards. The application is based on Contiki-Shell, an interactive text-based console that allows for definition of different command functions in the node, to be triggered locally over a serial USB or remotely from the gateway to one or multiple nodes. One of the limitations of Contiki-Shell is that when launching remote commands in a node, the node will not send back a response to the gateway. Thus, the Contiki-Shell core was modified to receive responses from the nodes when necessary. All wireless communications use the RIME stack, which provides a layered set of lightweight communication primitives. Although the system architecture is designed to be used in single-hop networks, it could be modified to work in multi-hop by using different RIME primitives. ContikiMAC is used for radio duty cycle, which keeps the radio off 99% of the time, saving power but increasing the communication delay.

The two modes of operation described in Section II, on-demand and periodic unattended, are supported through the remote shell commands. The periodic experiments can be any command available in the node, which will be called from the gateway with the period and other parameters. These parameters will be saved in a file in the memory, then read and executed when the node is powered on or after reset. The memory also stores the calibration file for the IMU as well as the data files collected from every experiment, using the Contiki file system. The main primitives of this file system are also implemented as shell commands, to be able to remotely read and write files, format the memory, etc.

Due to the performance constraints of the system, sampling high data rate sensors such as the audio and accelerometer at the same time while saving the data to the memory presents a challenge, which can be overcome by using the different resources already available in the microcontroller and IMU IC. To unburden the CPU from the microphone sampling, the ADC is used with DMA, and the blocks of data are stored in two ring buffer arrays in the microcontroller RAM. For the accelerometer sampling, a combination of the IMU IC internal buffer and another buffer in the microcontroller firmware is used. The file writing is also buffered in the external memory SRAM buffer before committing. This architecture ensures that both sensors can be sampled and saved at the required data rate without loss of frames.

### B. *Software and user interface*

To support the remote control of the nodes, a custom Python application is developed running on a laptop, which connects through the serial port to a gateway node. On top of the script, a PyQt-based graphical interface is built to facilitate the experiment management and node control as well as adding extra features. The gateway console tab handles the serial connection to the gateway, displays the full activity log, and allows for manually sending commands.

In Fig. 4 we see the network management tab, divided in three different areas. The top side contains controls for network discovery and a list of the nodes in range with information about the RSSI, Battery, Temperature, etc. for each individual node. From this list every node accelerometer and audio can be calibrated, as well as launching commands remotely. The bottom left shows the files in the memory of the selected node and allows to download individual files, as well as deleting and formatting the memory, while the bottom right lists the current periodic experiments set up on the node and permits adding, deleting, and stopping experiments.

Fig. 5 shows the data analysis tab, where we can display plots of the selected downloaded audio or accelerometer data in time (left side), as well as its power spectral density (right side), to quickly identify the peaks and frequencies of interest.

A 3D view tool is shown in Fig. 6, where we can load an STL CAD file of the target environment where we are deploying our network, as well as manually placing the individual sensor nodes in the exact position. This tool can be useful to study the environment, make deployment decisions, and aid in the development of propagation models by the automatic calculation of distances between nodes in 3D space, in contrast with other tools that only allow to arrange the nodes in a 2D space.

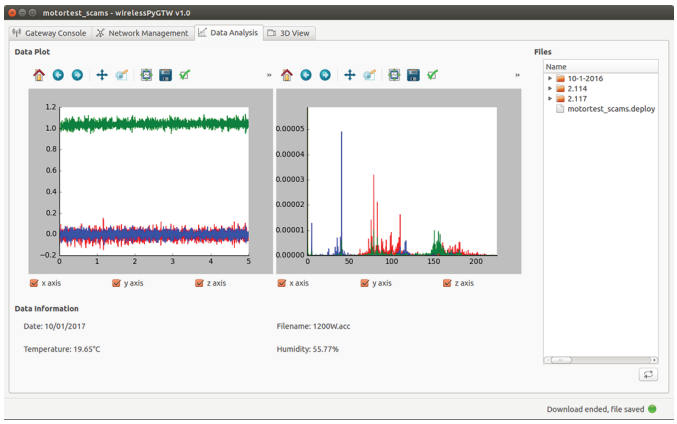



Fig. 5. UI Data view.

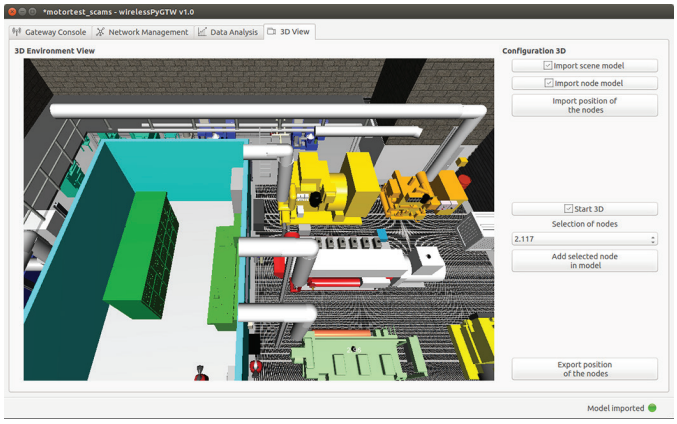



Fig. 6. UI 3D Environment view.

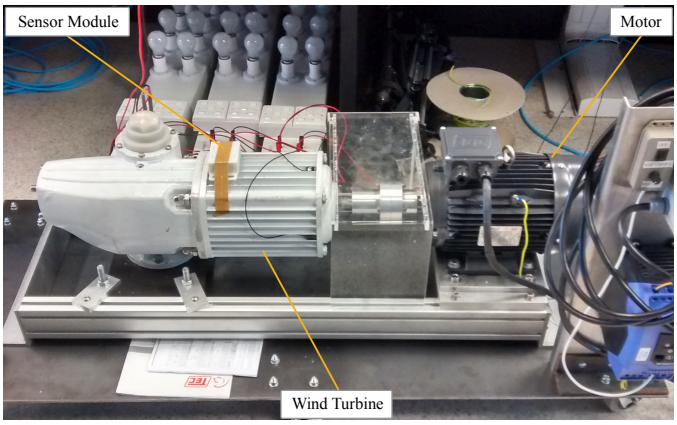



Fig. 7. Experiment setup of a motor connected to a wind turbine.

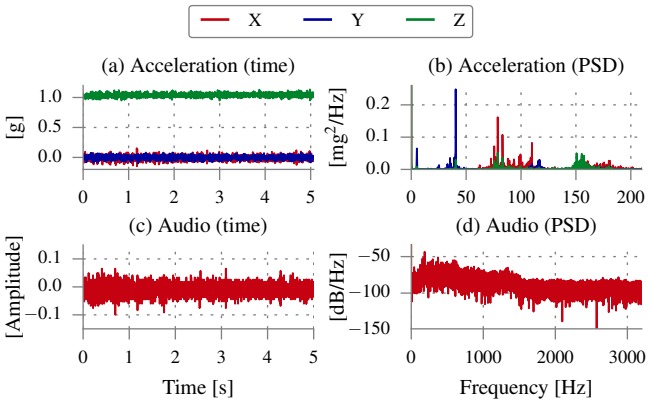



Fig. 8. Accelerometer and audio data collected at 1200 W load.

## V. Experimental Evaluation

### A. Three-phase motor connected to a wind turbine

To test the platform, we first attached one of the modules to a custom rig in the laboratory composed of a SWEA 2 KW wind turbine connected to a three-phase TEC 1500 rpm motor, driven by an Optidrive Variable Speed Drive (VSD). This setup can be seen in Fig. 7, with the node placed on top of the wind turbine with the z-axis of the accelerometer pointing up. The experiments were performed with the motor configured at 320 rpm and a variable load, set up from 300 W to 1800 W in 300 W increments. For each load configuration the sensor recorded 5 s of audio and accelerometer data.

As the motor and turbine were brand new and presented no flaws, there were no noticeable variations among the data collected for each experiment. Fig. 8a displays the accelerometer data for the 1200 W load test. Its power spectral density (Fig. 8b) shows a noticeable peak at 40 Hz for the y-axis, several peaks around 75-110 Hz for the x-axis, and a smaller magnitude spread around the 150-160 Hz for the z-axis. The audio data for the same experiment can be seen in Fig. 8c, with the energy distributed across the spectrum with no visible peaks (Fig. 8d).

### B. Single-phase motor connected to a gearbox

For the second experiment setup, we connected a single-phase WEG 1450 rpm motor to a reduction gearbox through a flexible coupling (Fig. 9), to be able to misalign the rotating axis by slightly displacing the motor attachment to the table. Due to the safety cage built on the setup, there was no space available on the top of the motor, therefore we placed the sensor module on the side, with the accelerometer y-axis aligned with the rotating axis. As the gearbox contained no load, the motor was run at full speed. Three different tests were run, first with the motor properly aligned with the gearbox, then misaligning it by 2 mm and 4 mm, sampling 5 s of audio and accelerometer data for each test.

From the collected accelerometer data, we observe a peak around the 94.5 Hz in all the tests for the three axes. Fig. 10 shows the power spectral density for the y-axis for the three experiments, where we can see how its magnitude approximately doubles for every misalignment step. The audio data showed a spectrum similar to the obtained in the previous setup.

For both experiment setups, the average time to download the audio was 3 min per file, while the accelerometer files took about 5 min due to the on-the-fly conversion from the binary

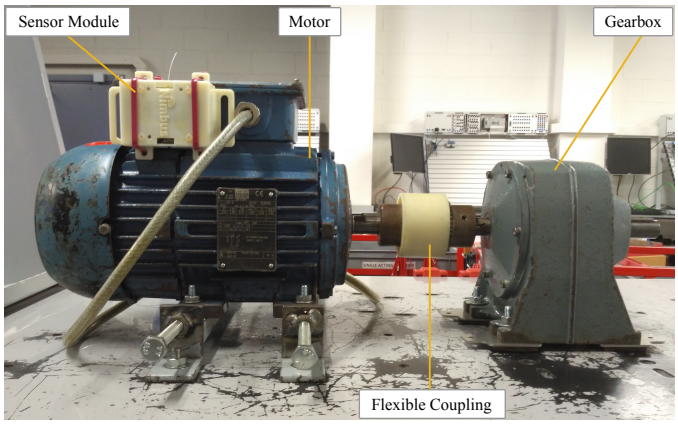



Fig. 9. Experiment setup of a motor connected to a gearbox.

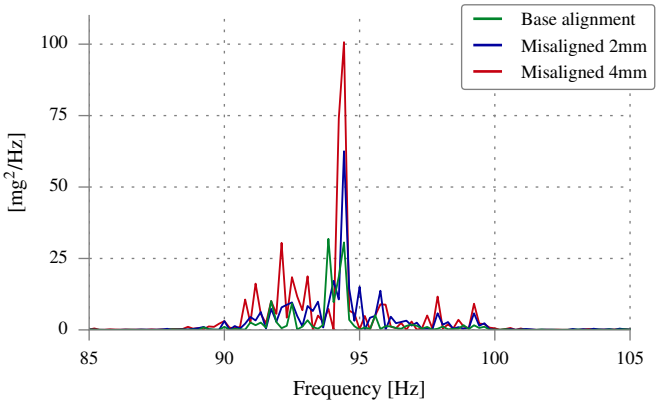



Fig. 10. Power spectral density of the y-axis accelerometer data.

data to the corrected values. Although the use of ContikiMAC as a radio duty cycle increases the download time, the much lower power consumption makes it a necessary trade-off to maximise battery life.

## VI. Conclusions and Future Work

In this paper, we presented an integrated WSN hardware-software platform for machine and structural monitoring. The custom module was based on a wireless IEEE 802.15.4-compatible microcontroller, with MEMS sensors for capturing vibration and audio data, and a humidity/temperature sensor for environmental conditions. The software tool allows the execution of on-site sensing experiments, as well as configuration of periodic autonomous experiments on each node. The included 3D view permits the visualisation of the environment and node placement, which can help when planning the network deployment in complex structures.

Although the experiments performed in both motor test rigs showed a reliable data collection, the long download times make it interesting to explore the feasibility of using compression algorithms or basic fault detection mechanisms on the node itself, to avoid sending a large amount of data that contributes to battery depletion.

Future work will include the implementation of several of these fault detection algorithms found in the literature, as an alternative to the collection of raw data files, and the effect of the use of these methods in the power consumption. In addition, further experiments in the wind turbine test rig will be done introducing flaws in the motor bearings to assess the capabilities of the system for fault diagnosis, as well as experiments with a full network of nodes in a real scenario.

## References

[1] MaREI - Marine Renewable Energy Ireland. [Online]. Available: http://marei.ie/ [Accessed Feb. 15, 2017].

[2] S. A. McInerny and Y. Dai, "Basic vibration signal processing for bearing fault detection," *IEEE Transactions on education*, vol. 46, no. 1, pp. 149–156, 2003.

[3] T. C. A. Kumar, G. Singh, and V. Naikan, "Effectiveness of vibration monitoring in the health assessment of induction motor," *International Journal of Prognostics and Health Management*, vol. 6, no. Special Issue Uncertainity in PHM) 007, pp. 1–9, 2015.

[4] S. Astapov, J. Preden, T. Aruvali, and B. Gordon, "Production machinery utilization monitoring based on acoustic and vibration signal analysis," in *Proc. 8th Int DAAAM Baltic Industrial Engineering Conf*, 2012, pp. 268–273.

[5] P. Henriquez, J. B. Alonso, M. A. Ferrer, and C. M. Travieso, "Review of automatic fault diagnosis systems using audio and vibration signals," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 44, no. 5, pp. 642–652, 2014.

[6] S. Chandrasekaran and T. Chitambaram, "Health monitoring of offshore structures using wireless sensor network: experimental investigations," in *SPIE Smart Structures and Materials+ Nondestructive Evaluation and Health Monitoring*. International Society for Optics and Photonics, 2016, pp. 980 416–980 416.

[7] A. Sabato, C. Niezrecki, and G. Fortino, "Wireless mems-based accelerometer sensor boards for structural vibration monitoring: A review," *IEEE Sensors Journal*, vol. 17, no. 2, pp. 226–235, 2017.

[8] X.-y. Xiong, F. Wei, J.-w. Li, M. Han, and D.-h. Guan, "Vibration monitoring system of ships using wireless sensor networks," in *Mechatronics and Automation (ICMA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 90–94.

[9] S. Schirrmacher, L. Overmeyer, and S. Lorisch, "Wireless condition monitoring of a marine gearbox," *Ship Technology Research*, vol. 63, no. 1, pp. 38–49, 2016.

[10] R. Severino, R. Gomes, M. Alves, P. Sousa, L. Ramos, R. Aguilar, E. Tovar, and P. B. Lourenço, "A wireless sensor network platform for structural health monitoring: enabling accurate and synchronized measurements through cots+ custom-based design," *IFAC Proceedings Volumes*, vol. 43, no. 17, pp. 375–382, 2010.

[11] J. Polastre, R. Szewczyk, and D. Culler, "Telos: enabling ultra-low power wireless research," in *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*. IEEE, 2005, pp. 364–369.

[12] A. Dunkels, O. Schmidt, N. Finne, J. Eriksson, F. Österlind, and N. T. M. Durvy. (2011) The contiki os: The operating system for the internet of things. [Online]. Available: http://www.contikios.org

[13] O. Basir and X. Yuan, "Engine fault diagnosis based on multi-sensor information fusion using dempster–shafer evidence theory," *Information Fusion*, vol. 8, no. 4, pp. 379–386, 2007.

[14] N. Baydar and A. Ball, "A comparative study of acoustic and vibration signals in detection of gear failures using wigner–ville distribution," *Mechanical systems and signal processing*, vol. 15, no. 6, pp. 1091–1107, 2001.

[15] C. Buschmann and D. Pfisterer, "isense: A modular hardware and software platform for wireless sensor networks," *6. Fachgespräch Sensornetzwerke*, p. 15, 2007.

[16] G. Mujica, V. Rosello, J. Portilla, and T. Riesgo, "Hardware-software integration platform for a wsn testbed based on cookies nodes," in *IECON 2012-38th Annual Conference on IEEE Industrial Electronics Society*. IEEE, 2012, pp. 6013–6018.