Hindawi
Journal of Sensors
Volume 2018, Article ID 2107679, 12 pages
https://doi.org/10.1155/2018/2107679

Research Article

A Sensor Network with Embedded Data Processing and Data-to-Cloud Capabilities for Vibration-Based Real-Time SHM

Nicola Testoni,[1] Cristiano Aguzzi,[2] Valentina Arditi,[1] Federica Zonzini,[1] Luca De Marchi,[1] Alessandro Marzani,[3] and Tullio Salmon Cinotti[2]

[1]*Department of Electrical, Electronic and Information Engineering, University of Bologna, Bologna, Italy*
[2]*Department of Computer Science and Engineering, University of Bologna, Bologna, Italy*
[3]*Department of Civil, Chemical, Environmental and Materials Engineering, University of Bologna, Bologna, Italy*

Correspondence should be addressed to Nicola Testoni; nicola.testoni@unibo.it

Received 31 December 2017; Revised 13 April 2018; Accepted 3 May 2018; Published 2 July 2018

Academic Editor: Christos Riziotis

This work describes a network of low power/low-cost microelectromechanical- (MEMS-) based three-axial acceleration sensors with local data processing and data-to-cloud capabilities. In particular, the developed sensor nodes are capable to acquire acceleration time series and extract their frequency spectrum peaks, which are autonomously sent through an ad hoc developed gateway device to an online database using a dedicated transfer protocol. The developed network minimizes the power consumption to monitor remotely and in real time the acceleration spectra peaks at each sensor node. An experimental setup in which a network of 5 sensor nodes is used to monitor a simply supported steel beam in free vibration conditions is considered to test the performance of the implemented circuitry. The total weight and energy consumption of the entire network are, respectively, less than 50 g and 300 mW in continuous monitoring conditions. Results show a very good agreement between the measured natural vibration frequencies of the beam and the theoretical values estimated according to the classical closed formula. As such, the proposed monitoring network can be considered ideal for the SHM of civil structures like long-span bridges.

1. Introduction

Real-time structural health monitoring (SHM) of civil and industrial buildings requires a network of smart sensors which are capable to simultaneously acquire signals coming from a plurality of transducers [1]. Meaningful information should then be conveyed to the final user either in a raw or in a processed form by means of easy-to-deploy communication infrastructures and stored in scalable and highly available database without compromising the performance.

Among the variety of monitoring approaches, techniques based on the changes of structural natural frequencies and modes shapes have been largely investigated and adopted in the civil engineering field spanning from historical buildings [2–5] to more modern long-span bridges [6–9]. The continuous analysis of vibration features, in fact, can provide extremely useful information about possible changes in the structural properties and in turn on the overall state of the structure.

As such, advanced sensing networks capable to be permanently installed on the structure to be monitored are currently under investigation. In particular, research developments strive to reduce (i) the weight and cost of the sensing elements, (ii) the sensor network power consumption, and (iii) the cost for the deployment of centralized data acquisition systems, as well as (iv) the amount of cables [10–13] whereas, in the other end, aimed at (v) handling and sharing large amount of collected data.

In this scenario, the use of microelectromechanical (MEMS) accelerometers, thanks to their low cost and low power features, allows to cope with drivers (i) and (ii) [14–17]. However, wireless sensors can neither store/process all raw data locally nor reliably forward the data: wireless technologies are in fact typically incompatible with high-rate and high-resolution data collection and long-term monitoring tasks [18, 19].

Recent advancements in information technologies made possible to integrate SHM with Internet to track sensible data anytime and anywhere: the concepts of Internet of Things (IoT) [20–23] and its most recent evolution, the Social Web of Intelligent Things (SWIT) [24], can be fruitfully applied to SHM to provide almost real-time comprehensive information about monitored structures to all the involved figures, thus allowing to tackle drivers (iii) to (v). At the same time, due to the amount and criticality of the collected data, problems in data collection quickly arise. These problems can be addressed by adopting database management systems designed to handle large amounts of data capable to provide high availability with no single point of failure. These systems are often decentralized, in order to be fault tolerant at a geographic level, scalable, and support replication to ensure data preservation under harsh conditions [25, 26].

In this work, a network of small footprint, low power, and lightweight MEMS-based sensor nodes for modal shape and natural frequency analysis is presented. Network components are an upgraded and repurposed version of a previous work [27] aimed at guided wave detection. The network was designed to be, at the same time, low cost, scalable, and easily reconfigurable to be suitable both for short-term and long-term monitoring tasks. Weight and size reduction in the presented network are crucial to reduce cabling complexity, ease deployment, and demonstrate that a functional SHM system comprehensive of data collection and signal processing can be permanently embedded in structures without negatively impinging on their weight and size.

To achieve this goal, the proposed network replaces the traditional point-to-point connection between sensors and acquisition unit with the sensor area network (SAN) bus based on data-over-power (DoP) communication. To optimize this shared communication resource, techniques similar to that employed in a wireless sensor network can be used, yet sensor node power consumption is much less of a concern, allowing for continuous, real-time sampling of the structure health status. As a result, a sensor near electronics, which allows to preprocess monitoring information, losslessly encode it and transmit it to the cloud via a purposely designed companion gateway device was developed.

This sensor node was designed to be connected to other similar devices to create a network of up to 64 nodes: the total number of nodes on the network can be expanded arbitrarily by means of repeater nodes not discussed in this work. Communication to a notebook PC or to an embedded platform like Raspberry is performed in a half-duplex fashion at an effective speed of 200 ksps by means of a low-voltage, high-speed, half-duplex RS485 transceiver.

The paper is organized as follows. In Section 2 the sensor node, the gateway, and the implemented database interface are discussed; these materials are used as building block for the SHM sensor network. Section 3 discusses sensor calibration and data acquisition, and processing and collection. Finally, in Section 4, after discussing the application of the proposed sensor network to a case of study, the results of the modal shape and natural frequency analysis algorithms applied to the acquired dataset are compared with the theoretical predictions. Conclusions are then drawn in Section 5.

2. Materials

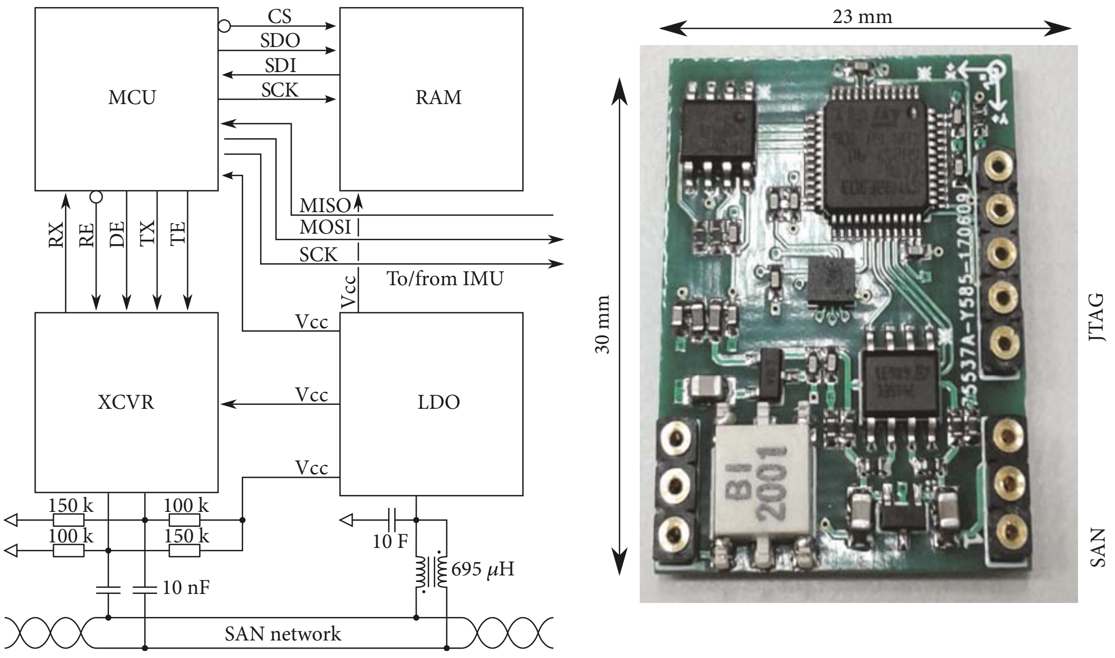
2.1. Sensor Node. The sensor node is built upon a four building block architecture as shown in Figure 1. All the devices in the sensor node are capable to withstand temperatures in the range of −40°C to 85°C.

At its heart is an ST Microelectronics STM32F3 32 bit, 3.3 V low power microcontroller unit (MCU): it belongs to a family of low-voltage 32 bit mixed-signal MCUs with DSP and FPU instructions which feature an integrated 8 MHz oscillator, a 40KiB SRAM, and a 256KiB FLASH, thus reducing the number of external components, hence the area occupied. The two integrated serial peripheral interface (SPI) and universal synchronous/asynchronous receiver/transmitter (USART) contain all the clock generators, shift registers, and data buffers necessary to perform input/output serial data transfers independently of device program execution. The maximum current consumption in active mode at a system clock frequency of 64 MHz is less than 40 mA but it can be decreased to less than 1.37 μA in standby mode.

In order to expand its limited memory capacity, a microchip 23LC1024 128Kibit Serial SRAM device is used as a temporary storage for the processed data samples. The MCU can access it through a dedicated SPI interface. It features an unlimited number of read/write cycles and zero write time, allowing for data rate up to 20 Mbps in sequential access mode. Still, it has a very low power consumption; when in standby, only 2 μA is absorbed from the 3.3 V power supply, while typical operative current is less than 2 mA.

To interface the MCU to the SAN network, an ST Microelectronics ST3485EB 3.3 V low power transceiver (XCVR) for RS-485 and RS-422 communications is used. All transmitter outputs and receiver inputs are protected to ±15 kV IEC 610004-2 air discharge. The MCU can access it through a dedicated USART interface. The driver is short-circuit current limited and is protected against excessive power dissipation by thermal shutdown circuitry that places the driver outputs into a high impedance state. It features a guaranteed 12 Mbps data rate at a very low power consumption; when in current shutdown mode, only 1 μA is absorbed from the 3.3 V power supply, while typical operative current is less than 1.5 mA. A mesh of passive components is used to interface the XCVR to the SAN network: two 10 μF capacitors are used to capacitively couple the XCVR input/outputs to the bus, while a resistive partition network allows for fail-safe biasing at less than 27 μA.

The entire sensor node is powered through a Texas Instrument LM3480 100 mA low-dropout (LDO) regulator. A device with a fixed 3.3 V output voltage was selected to reduce the number of required external components. This IC features operation from an input voltage as high as 30 V and an ensured maximum dropout of 1.2 V at the full 100 mA load, which is compliant with the 5.0 V power supply of the entire SAN bus. Its quiescent current is less than 1.75 mA and is inversely proportional to the load current. A



(a)

(b)

Figure 1: Schematic diagram of the sensor node (a) and the relative prototype (b).

695 μH differential mode choke is used along with a 10 μF capacitor to extract power from the SAN bus and feed it to the LDO.

Finally, the sensor node is customized to collect accelerations and angular velocities by means of an ST microelectronics LSM6DSL iNEMO inertial measurement unit (IMU). This device is a system-in-package featuring a 3D digital accelerometer and a 3D digital gyroscope performing at 0.65 mA in high-performance mode and enabling always-on low power measurements. The MCU can access it through a dedicated SPI interface. It has a full-scale maximum acceleration range of ±16 g and a maximum angular rate range of ±2000 dps. When in current shutdown mode, only 3 μA is absorbed from the 3.3 V power supply.

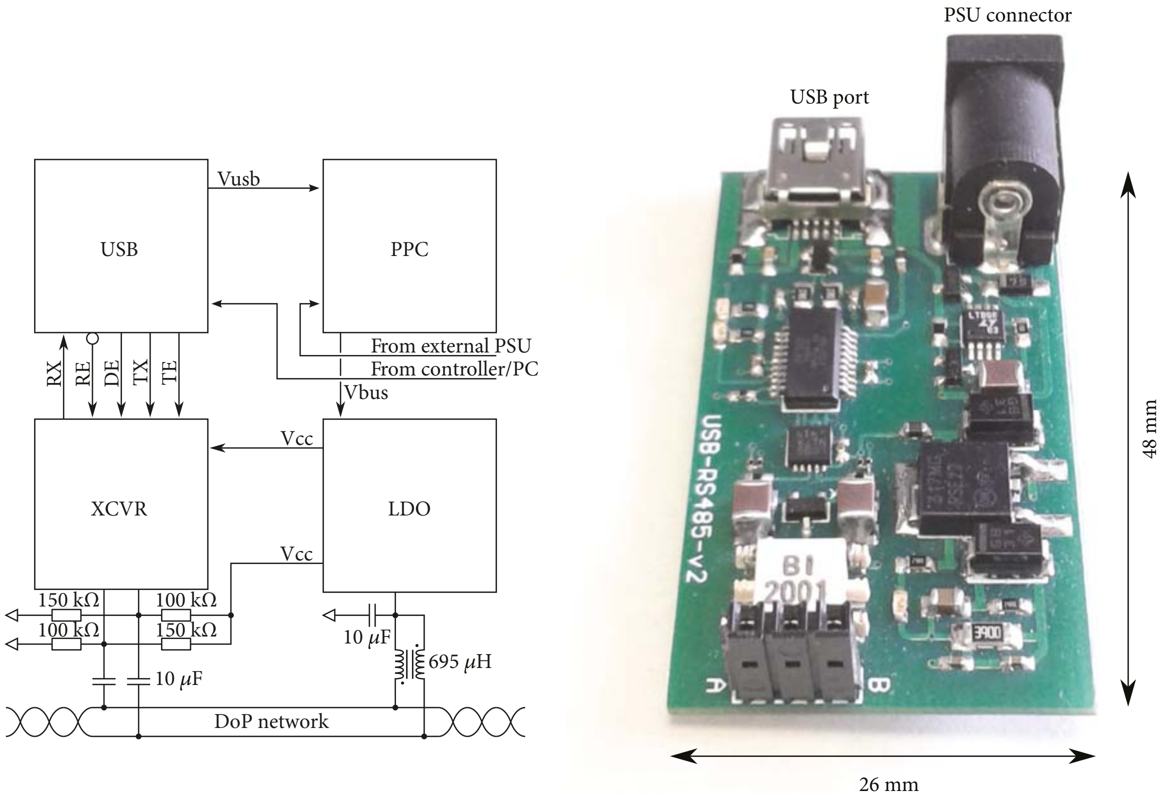
The chosen triaxial MEMS accelerometer, with a sensitivity of 825 mV/g in a small package (2.5 mm × 3.0 mm × 0.8 mm) compares well versus commercial uniaxial high sensitivity piezoelectric accelerometers with similar sensitivity (1000 mV/g) in a much larger package (48 mm × 48 mm × 37 mm), as well as triaxial high sensitivity piezoelectric accelerometers with lower sensitivity (100 mV/g) in a compact package (14 mm × 14 mm × 14 mm); both these solutions do not offer signal processing capabilities. As such, the proposed solution demonstrates the possibility to integrate within a small package both triaxial sensing and signal processing capabilities.

Structural vibrations are recorded by the IMU and stored in the MCU internal RAM. Program instructions and filter coefficients are stored in the MCU embedded flash memory for the purpose of data processing. After processing, data can be stored in the external SPI serial SRAM, waiting for remote retrieval and collection.

The mean value of the total current consumed by a woken-up sensor node amounts to 44.8 mA when the SAN bus is powered at 5.0 V, whereas a sleeping node consumes less than 2 mA. Each sensor node is roughly 30 mm × 23 mm wide and weighs less than 5 g, making it attractive for fields where size, power, and weight reduction are crucial.

2.2. Gateway Device. A Raspberry Pi is employed as a smart data relay. This device is a low cost, flexible, single-board computer with a small form factor which is popular in the IoT field and starting to get attention from the SHM community [20, 21, 28]. The Pi comes in various models (in date publication order): A, B, A+, B+, 2, Zero, 3, and Zero W. Among them, the current flagship board of Raspberry Pi Foundation, the Raspberry Pi 3(https://www.raspberrypi.org/products/raspberry-pi-3model-b/) was chosen. The board is equipped with a quad-core 1.2GHz Broadcom 64 CPU, 1 GB of RAM and BCM43438 wireless LAN interface. It mounts a Debian-based Linux distribution called Raspbian. Its form factor (85 mm × 56 mm) is not as small as the nodes in SAN but it can still be suitable in SHM without any disadvantages, considering that most likely there will be only one gateway in a monitored structure. About the power consumption, the device also drains considerably higher currents than the sensor nodes. In idle state, as referred in [29], it consumes 260 mA, whereas at runtime operation, current consumptions raise to 480 mA. This difference compensates for the computational and communicational capabilities gain.

The main task of the Pi is to send the sensed data to an online database and serve as an interface to allow remote communication with every SAN node. The adoption of the Pi 3 has cut the cost of development thanks to its easy programming and connectivity interfaces.



(a)

(b)

Figure 2: Schematic diagram of the SAN interface (a) and the relative prototype (b).

The device is connected to the SAN through ad hoc serial interface and communicates with remotes nodes using its Wi-Fi interface. The physical SAN network interface is built upon a four building block architecture as shown in Figure 2. All the devices in the network interface are capable to withstand temperatures in the range of −40°C to 85°C.

An FTDI FT231X USB to full handshake UART integrated circuit (IC) is used to provide USB connectivity to the Raspberry. The entire USB 2.0 full speed protocol is handled on this chip, allowing for data transfer rates from 300 baud to 3 Mbaud at TTL levels. It is operated by a single supply line taken directly from the USB bus and internally regulated to by integrated low-dropout (LDO) regulators. When active, it typically consumes 8 mA, but it can also enter a suspend state in which power consumption drops to 125 μA.

A Linear Technology LTC4414 low loss PowerPath™ Controller (PPC) is used to control an external P-channel MOSFET in order to create a near ideal diode function for power switchover. This permits the interface to perform a highly efficient management of the available power sources, namely, the 5.0 V, 500 mA USB power supply line, and an external 3.5 V to 36 V, 5A auxiliary power supply unit. When conducting, the measured voltage drop across the MOSFET is typically 20 mV with a typical $R_{DS(ON)}$ of 31 mΩ. The load is automatically disconnected from the USB power line when the auxiliary source is connected. The wide supply operating range of the PPC also supports operations from one to eight Li-ion cells in series to manage battery-operated scenarios. The low quiescent current (30 μA typical) is independent of the load current.

The PPC primarily feeds a Texas Instrument LM317 medium current 1.2 V to 37 V adjustable LDO voltage regulator. When powered through the 5.0 V USB source, quiescent current consumption never exceeds 1.5 mA; conversely, currents up to 4.0 mA can be measured when a higher voltage is supplied through the auxiliary power line. Similarly to the sensor node, the LDO feeds an ST Microelectronic ST3485EB RS485/RS422 transceiver (XVCR), used to interface the UART to the SAN network. Through the LDO section, the SAN network bus is also powered by means of two mutually coupled 695 μH inductors in an antiparallel configuration. This filters out common mode interferences from the SAN network bus, while allowing the flow of power supply current [30].

The mean value of the total current consumed by the interface circuit mounts to 12 mA when the SAN bus is powered at 5.0 V, whereas a sleeping interface consumes less than 2 mA. Each interface node is roughly 48 mm × 26 mm wide and weigh less than 10 g, making it attractive for fields where size, powe,r and weight reduction are crucial.

Concerning power consumption, one of the gateways is about that of 11 sensor nodes: a system comprising 16 nodes and one gateway device powered at 5 V consumes slightly less than 6 VA. This compares very favorably with a commercial system capable to collect data from the same amount of node,

Table 1

SensorId	Value	Timestamp
A40CFF12..	5	1514454710
A40CFF12..	4.2	1514454709
A40CFF12..	4.1	1514454708
A40CFF12..	4.1	1514454707
A40CFF12..	3	1514454706

Partition key: *((sensorId), value*, and *timestamp).*

which requires more than 300 VA for data acquisition without offering data processing capabilities.

2.3. Online Database. Data collected in loco by sensors in a monitored structure should be deployed online to implement a remote monitoring system. Furthermore, logs about events occurred during the structure lifetime can provide useful information for planning maintenance operations. Consequently, in our software architecture, we employed a database to store this data. The database technology is a Cassandra (http://cassandra.apache.org/) instance installed in a remote data center.

Cassandra is an open source NoSQL database management system maintained by the Apache foundation. It offers robust support for clusters and it is designed to handle large amount of data. Furthermore, it features high availability due to its distributed nature with no single point of failure [31]. Studies have also shown its scalability and high throughput capabilities [25]. It employs a tailored data model to achieve its performance. The most notable difference from relational databases [32] is how tables are structured. Specifically, how rows are indexed in the tables and how tables are stored throughout the cluster which allows to fastly insert, delete, update, and read data.

The *primary key* is subdivided in two sections. One defines the *partition key* that identifies data locality, which means rows with the same *partition key* are stored inside the same cluster. The other, the so-called *clustering columns*, specifies how the rows are sorted, which allows to fast insert, delete, update, and read data. The combination of *partition key* and *clustering columns* should be unique to obtain a correct data model.

For example, consider to record acceleration values along the *x*-axis of an accelerometer and to query the database for its top five values as shown in Table 1.

The *sensorId* column is the *PARTITION KEY* ensuring that all the data from the sensor A40CFF12.. are stored in the same node. Then the rows are ordered by *value* (descending order) and finally by *timestamp* (also in descending order) that guarantee uniqueness. A query issued to the database could be

```
SELECT * FROM acceleration by sensor
WHERE id = 'A40CFF12..'          (1)
LIMIT 5
```

Accordingly to this query, the database engine will search for the node where the id is stored, then thanks to the already ordered rows retrieve the first five.

The features of Cassandra are perfectly suited for the implementation of structural monitoring systems. In particular, its high throughput allows to manage high rates of events generated in critical scenarios (i.e., earthquakes) and high sampling frequencies. High availability and robustness guarantee data collection even in presence of failures. Finally, the capability to handle big amount of data is crucial for large structures which can be sensorized with hundreds of devices.

For the purpose of this work, we deployed a Cassandra 2.2 instance in our testing environment on an Arch Linux 4.14.6-1 single cluster with default settings.

3. Methods

3.1. Sensor Calibration. Accelerometers used in this work, the ultra-low power high-performance three-axis iNEMO by STMicroelectronics, are guaranteed by the producer to be factory calibrated. It means that trimming values are stored inside the device in a nonvolatile memory, downloaded into registers, and used to correct any measure during the activity. However, in order to obtain data as much precise and reliable as possible, we adopted an additional calibration procedure suitable to be performed by the end-user on the field [33–35].

The calibration procedure starts with the acquisition of six values corresponding to the acceleration vector completely projected on every coordinate axis, in both positive and negative directions (e.g., vector (0,0,1) for positive direction of the *x*-axis). Each acquisition lasts a few seconds, then the values are averaged and normalized by the MCU and a single vector is associated with the corresponding stationary position. Comparing the raw measurements (gathered in matrix **W**) to ideal values (theoretical stationary positions, in matrix **Y**), the calibration parameters are calculated as the twelve coefficients that form the so-called calibration matrix (**X**), so that

$$\mathbf{Y} = \mathbf{W}\mathbf{X}. \quad (2)$$

Finding **X** means solving an overdetermined linear system and this can be done through the evaluation of pseudoinverse matrix

$$\mathbf{X} = \left[\mathbf{W}^{\top}\mathbf{W}\right]^{-1}\mathbf{W}^{\top}\mathbf{Y}, \quad (3)$$

when all the columns of **W** are independent, as in this case. The obtained calibration parameters are stored on the sensor node flash memory and used to correct every new raw acquisition, until a new calibration procedure or a reset command is sent.

3.2. Data Acquisition. The whole system is able to measure projections of acceleration vectors on the three space directions x, y, and z with different features. Accelerometers were programmed to work, according to specific needs of the users, in a selectable full-scale of ±2 g, ±4 g, ±8 g, or ± 16 g, from a minimum sampling frequency of 1 Hz up to 5.3 kHz and in three different operation modes (high, normal, and low power) that correspond to various data resolutions in bits (12, 10, and 8, resp.). Moreover, to further decrease power consumption, when the device does not collect data, the sensor node is automatically set to a default condition of power-down mode that lasts until a specific command is sent.

The accelerometer embeds a 10 bit, 32-level FIFO buffer for each of the three output channels. This resource allows a consistent power saving because MCU does not need to poll for new data frequently and can be put to sleep mode for a much longer period of time. Due to the capacity of 10 bits for each slot in the stack, when the FIFO is operating, the only data resolutions possible are normal mode and low power mode.

More specifically, data can be downloaded from the accelerometer output registers according to four different working modes of their FIFO buffers: (i) bypass mode, (ii) FIFO mode, (iii) stream mode, and (iv) stream-to-FIFO mode.

In the *bypass mode*, the FIFO is nonoperative: the output registers used by the accelerometer are just six and they always contain the current data.

In *FIFO mode*, data from x, y, and z channels are stored into the FIFO, which is continuously fed until it is full. When all the levels are occupied, the buffer stops collecting new data. The device can generate an interrupt signal when FIFO reaches a given filling (watermark) level, in order to avoid data loss.

The *stream mode* differs from the previous one because when the buffer is full new data overwrite the older ones, even if they have not been read yet. Watermark level handling is applicable in this case too.

Finally, in the *stream-to-FIFO mode*, the transition from stream-to-FIFO mode is provided when a trigger event occurs.

To improve the efficiency of the whole system and reduce the CPU time, all the reading stored data operations were created to be handled by interrupt routines. In this work, the stream mode was chosen because it gives to the user more flexibility concerning the sampling frequency while guaranteeing the availability of fresh acceleration data.

3.3. *Parameter Estimation.*

Modal shape and natural frequency estimation were conducted in this work in two modalities: on-sensor and off-sensor.

3.3.1. *On-Sensor Estimation.*

Modal estimation was developed on-sensor by implementing procedures to detect relevant peaks in the acquired signal spectrum.

More precisely, we exploited functions from CMSIS DSP library to perform fast Fourier transform for floating point, 32-bit data. The magnitude of the complex Fourier transform coefficients is passed as input to an algorithm of peak detection [36]. Such algorithm compares the magnitude of the ith coefficient ($X[i]$) with a given threshold, which can be set independently for each axis and to the previous and following ones, $X[i-1]$ and $X[i+]$, respectively: if $X[i]$ is greater than threshold and its adjacent samples, the peak is detected.

In order to avoid false natural frequencies detection, during the experimental trials, many different values for the threshold levels in the peak detection algorithm were tested. According to the capability of the system to act differently on each axis, it was also possible to optimize the sensor node sensitivity for each channel separately. Moreover, to identify the peak position with more accuracy, a correction factor to the identified quantized frequency i was applied by performing a quadratic interpolation:

$$i - \frac{|X[i+1]| - |X[i-1]|}{2(|X[i+1]| - 2|X[i]| + |X[i-1]|)}. \tag{4}$$

Such correction is applied to all the identified peaks as part from the zero frequency acceleration which is related to gravity.

3.3.2. *Off-Sensor Estimation.*

More sophisticated algorithms are computed off-sensor to extract natural frequencies and modal shapes. It is worth noting that the extraction of the modal parameters in the pristine structure is of extreme importance because it creates a set of benchmark values that must be as accurate as possible to detect abnormalities during the real-time estimation stage. Several procedures were considered to compute the probability density function (PSD) of the sensor acquisitions either based on parametric or nonparametric approaches. Among the algorithms of the first group, we considered autoregressive (AR) [37] and AR + noise models [38] which are suited for operational modal analysis in presence of strong acquisition noise. As concerning the nonparametric approaches, we have considered the frequency domain decomposition (FDD) [39], periodogram, and Welch estimation [38] methods.

It is worth noting that, besides data and power communication, the data-over-power bus also natively allows for data acquisition time base synchronization and consequently the output-only estimation of modal shapes. According to the authors in [40], the synchronization of data recording has a direct impact on the covariance index of the original mode shape to its time-shifted version estimated on-sensor. In the proposed sensor network, the maximum delay between each sensor is 50 μs, which corresponds to a maximum degradation of 1% in the correlation factor of the fourth mode.

Algorithms in the time and frequency domain were developed for the purpose of output-only estimation of modal shapes, followed by the application of the second order blind identification (SOBI) method, a strategy which reveals the independent components hidden within a set of measured signal mixtures [41]. Alternatively, the frequency domain decomposition (FDD) method can be used to identify modal parameters of a dynamic system by applying the singular value decomposition (SVD) technique to the output spectral density matrix, whereas the time domain decomposition (TDD) computes the SVD of the energy correlation matrix referred to bandpass-filtered data around the expected vibrational modes [39].

3.4. *Data Collection.*

As anticipated in Section 2.2, the Raspberry Pi 3 device is in charge of data gathering. A custom made Go language software [42] able to query every node in the SAN network and send the data to the remote Cassandra service is installed on board. The process is running in the background as a service in the Raspbian OS and behaves as described in Algorithm 1.

```
procedure startSensor(address: int)
    command ← new AlwaysRunCommand
    send(address,command)
    waitForAck(address,TIMEOUT)
Init loop:
for all sensor address do
    startSensor(address)
Main loop:
for all sensor address do
    command ← new GetAccCommand
    send(address,command)
    data ← read(TIMEOUT)
    ifvalid(data) then
        send(address,ACK)
        acc ← decode(data)
        timestamp ← decode(data)
        sendToDB(address,acc,timestamp)
    startSensor(address)
gotoMain loop.
```

Algorithm 1: Data collection algorithm.

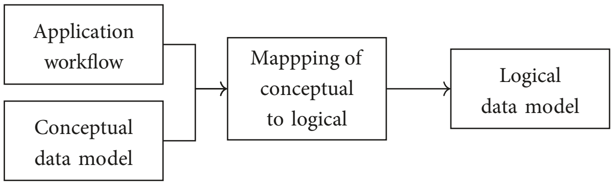


Figure 3: Query-driven design process phases.

First, all sensors are programmed to start the sensing phase in the *Init loop*. The *startSensor* procedure sends to a specific address the *AlwaysRunCommand* that defines the triggers in every spatial axis of acceleration sensing space. Each sensor can perform both individual and collective triggering, hardware, and software; in particular, each sensor is capable to receive and transmit trigger information from and to the other sensors in the network with a maximum delay of 50 μs, which is less than the minimum sampling period of 189 μs at 5.3 KHz. If the acceleration sensed in one of the sensors exceeds a given threshold, data is stored in the FIFO buffer and the event is communicated to all the other sensors. This event triggers the *Main loop* where data is retrieved from the buffers in the sensors. After data reading, the sensor must be reprogrammed to sense the acceleration continuously. Finally, the primitives *send* and *read* manage the serial connection with the SAN while *sendToDB* function communicates with the Cassandra cluster over the Wi-Fi interface.

3.4.1. Data Model Design. Using Cassandra as database implies the use of a new data model design created to ensure sound and efficient design [43]. This kind of development process is called query-driven design, and it is based on the definition of tables from the application workflow. As shown in [43], the process is made up of several phases, depicted in Figure 3.

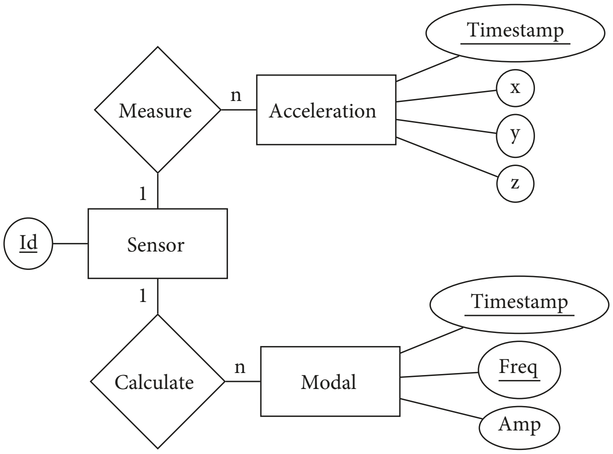


Figure 4: Logical data model of the monitoring application; primary and clustering keys are represented by underlined text.

Table 2: Tables for the queries in the monitoring application: entries marked with K representing the *primary key*, whereas C stands for the *clustering key*.

Q1: AccBySensor	
SensorId	K
Timestamp	C
Value	
Q2: AccBySensorTH	
SensorId	K
Value	C
Timestamp	C
Q3: ModBySensor	
SensorId	K
Freq	C
Timestamp	C
Amp	
Q4: ModAccBySensor	
SensorId	K
Timestamp	C
Freq	C
Value	
Amp	

In the *application workflow*, we start by defining the queries needed for adequate application behavior.

(Q1) Retrieve acceleration data produced by a sensor in a certain time interval (latest-first).

(Q2) Retrieve acceleration data produced by a sensor above a certain threshold (latest-first).

(Q3) Retrieve modal parameters estimated by a sensor in a certain time interval (highest frequency first).

(Q4) Retrieve combined acceleration and estimated modal parameters (highest frequency first).

These are basic queries that allow application users to get information about the monitored structure. Q1 ensures data enumeration (i.e., list all data acquired by SENSOR 1),

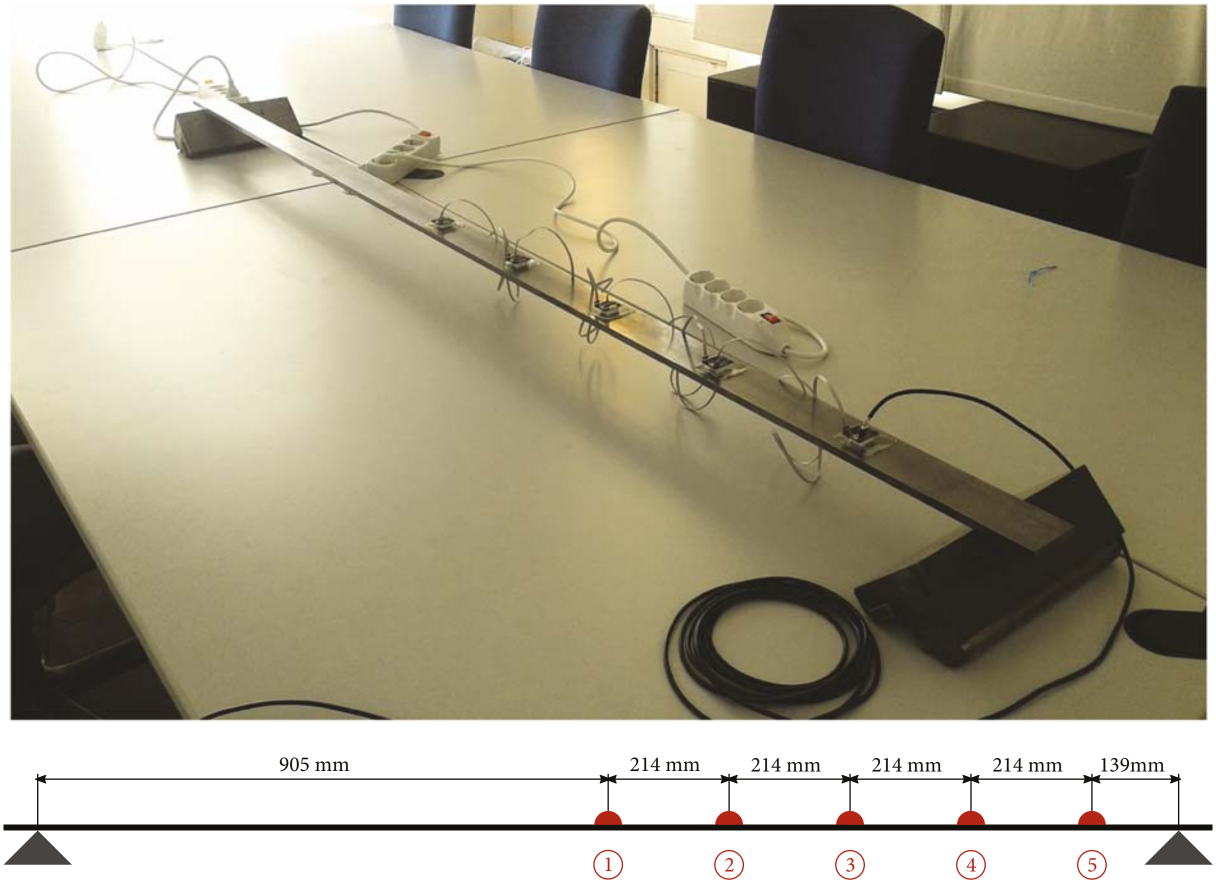


Figure 5: Experimental setup for modal shape and natural frequency verification and comparison.

Q2 gives a basic level of filtering (i.e., select only relevant accelerations), Q3 concerns modal parameter data enumeration, and finally Q4 correlates an acceleration value with modal parameters at that timestamp.

Combining the defined queries with the logical data model for the monitoring application shown in Figure 4 and following the rules defined in [43], the tables reported in Table 2 are crafted, which guarantee the correctness of data model stored in the Cassandra database. Notice that if a new kind of sensor data is to be gathered, the system can be scaled by adding a new pair of tables satisfying Q1 and Q2.

4. Result and Discussion

A network comprising five sensor nodes and one interface connected in a daisy-chain fashion was tested in a setup which consists of a simply supported $L = 1900$ mm steel beam with cross-section base $b = 60$ mm and height $h = 10$ mm in free vibrations conditions, as shown in Figure 5. Sensors node are separated by 214 mm with respect to the first sensor, which is placed close to the beam center.

The beam was excited in correspondence of its center by mean of an impact hammer, thus allowing it to oscillate in condition of free vibrations. The amplitude of the acquired signals ranged from 0.1 g to 2.3 g, respectively, referred to the minimum and maximum value of the recorded induced acceleration along the z-axis. The chosen impact position is very close to the antinode of the first vibrational mode: as a result, this mode is much more energetic than other recorded modes, showing a difference of nearly 40 dB which can intensely affects the optimal performance of modal shapes reconstruction techniques.

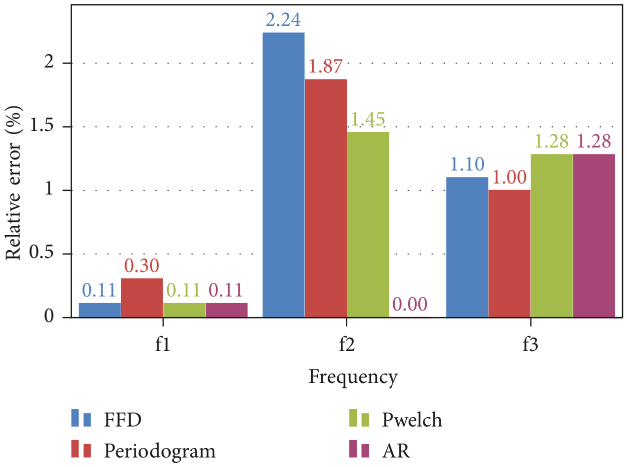


Figure 6: Error distribution in vibrational modes extractions comparing different techniques of PSD estimation.

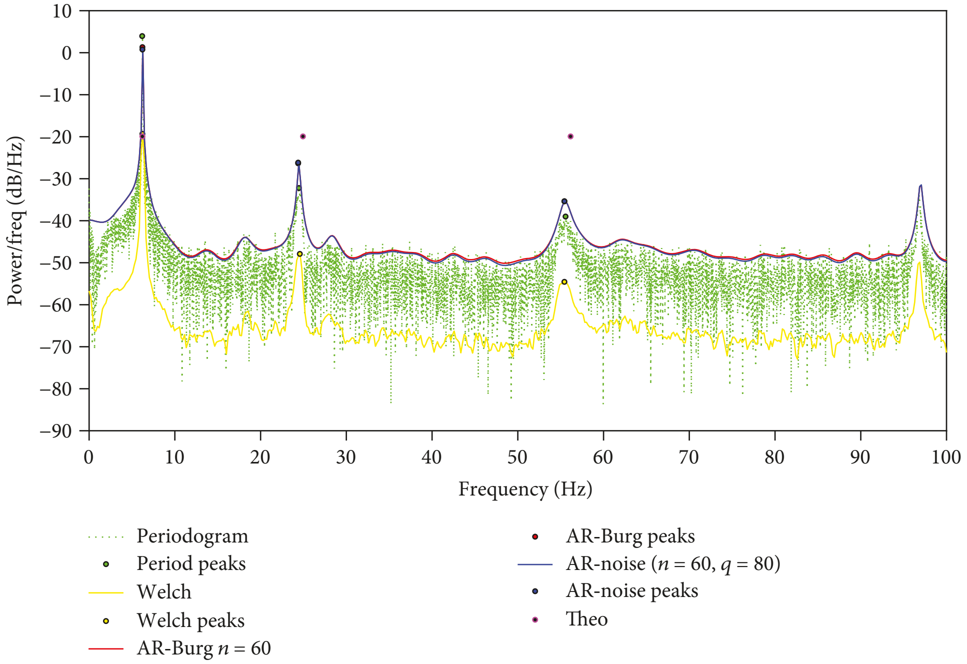


Figure 7: Comparison between different PSD estimators.

Signal processing techniques have been applied to the raw acquired signals to extract natural frequencies and modal shapes. The results of the modal shape and natural frequency estimation algorithms applied to the acquired dataset were compared with theoretical predictions obtained from the physical model of the beam to assess the performance of the implemented circuitry. In particular, the extracted first three natural frequencies are compared with the nominal values estimated as 6.25 Hz, 24.99 Hz, and 56.24 Hz through the theoretical formula:

$$f_n = \frac{1}{2\pi}\frac{(n\cdot\pi)^2}{L^2}\sqrt{\frac{E\cdot I}{\rho\cdot A}}, \tag{5}$$

where $\rho = 7880\,\mathrm{kg/m}^3$ is the density of steel, $A = b\times h$ and $I = bh^3/12$ are the cross-section area and moment of inertia, respectively, and n is the frequency index. Here, for accurately measured beam dimensions and weight, a value of Young's modulus $E = 195\mathrm{GPa}$ was assumed to minimize the error on f_1.

As it can be seen in Figure 6, the outcome of the algorithms computed is highly consistent with respect to the physical model, even in presence of high differences in the energy distribution. The relative error in natural frequencies estimation is always below 2.5% even in the worst case, related to the middle frequency. At low frequency, the performance of the spectral estimation methods is almost the same, whereas autoregressive techniques are more accurate for the estimation of the second vibrational component. At higher frequencies (above 30 Hz), there are no relevant differences among parametric and nonparametric algorithms.

Considering the fact that AR models are suited for systems affected by noise, the obtained results suggest that it is necessary to select the spectral estimation methods depending on the SNR. As shown in Figure 7, the PSD computed with the techniques mentioned demonstrates that, at low frequencies, there is an evident vertical alignment between computed and theoretical values, whereas the output is less precise when frequency increases. It is also evident that even the fourth natural frequency can be estimated from each of the tested algorithms, as testified by the peaks around 97.0 Hz.

Coherently with the literature [44], SOBI technique does not perform equally well as FDD or TDD since it is suited for dealing with signals sources in which mode strengths are each other similar and excitation is stationary. For real scenarios, characterized by low SNR values, noise can deteriorate the performance of the SOBI approach, especially while considering that no prior preprocessing is required apart from whitening raw signals.

The impact of sampling frequency on performance is also not negligible: in fact, by reducing the sample time, the number of samples to be used for modal shapes reconstruction is correspondingly augmented. This condition is of fundamental importance when some of the expected natural frequencies of the structure under test are close to Nyquists frequency: in such a scenario a strong correlation between estimation errors and sampling conditions is expected [45].

Results depicted in Figure 8 display the first three mode shapes extracted with three different techniques superimposed to the theoretical mode shapes. Again, the experimental behavior of the structure very well fits the physical model at low frequency, whereas deviation is more remarkable for modes related to higher natural frequencies.

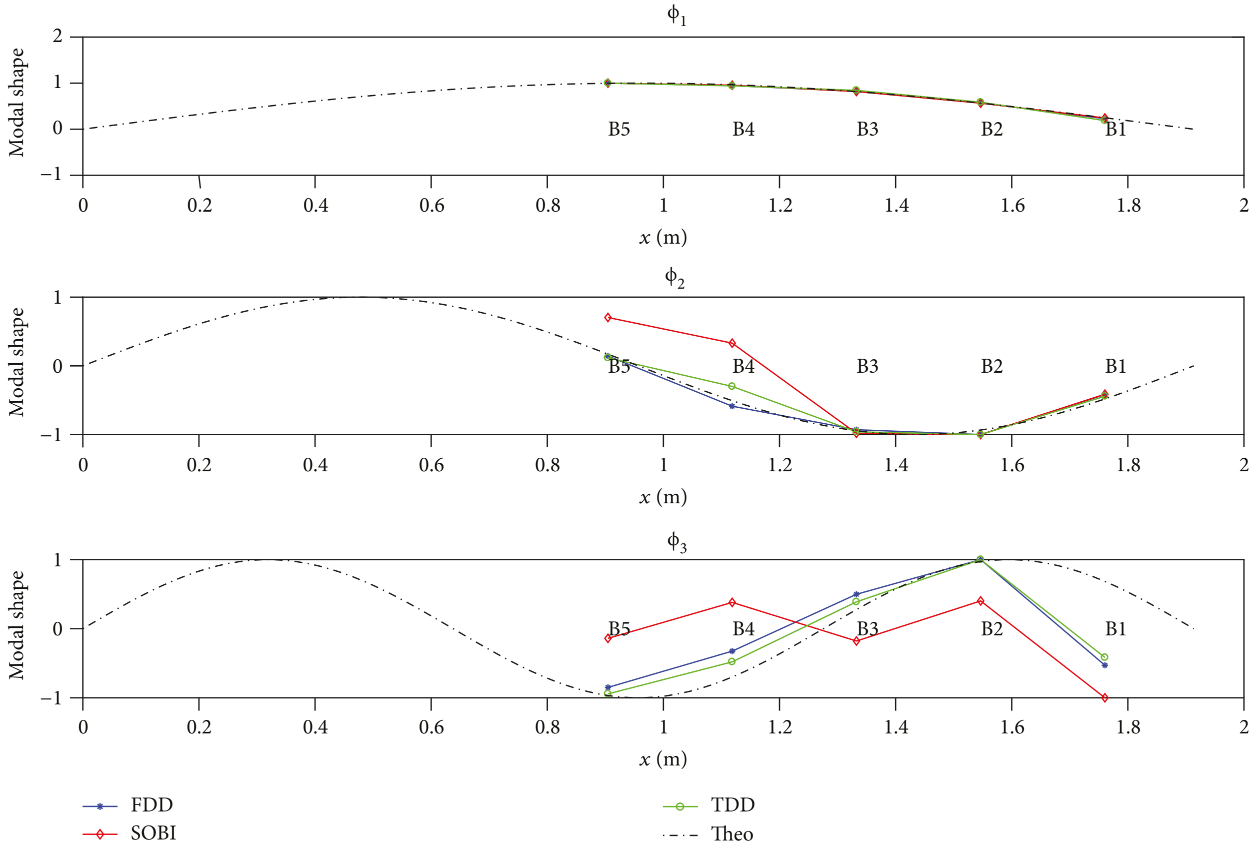


Figure 8: Modal shapes extracted with different techniques: FDD, TDD, and SOBI.

Such behavior is due to the fact that the lowest frequency component is the most energetic. It is also worth highlighting that, although the SOBI method is totally unsupervised, its performance is almost equivalent to FDD and TDD, making it suitable for autonomous damage detection systems.

5. Conclusions

This work reports on the implementation of a lightweight SHM sensor network comprising miniaturized sensor nodes and a gateway based on a single board PC. The sensors and the gateway are capable to exchange data and power on the same bus, thus minimizing the burden due to cabling. Also, by exploiting a wired connection, power figures and data bandwidth are fully compatible with the requirements of real-time data acquisition and processing. Two different processing modalities were made available, one exploiting the data processing capabilities of the sensor nodes, the other the ones of the gateway itself. Data processing results can be uploaded to a highly available database system for data collection and further interpretation without compromising the overall system performance. A modal shape and natural frequency estimation experiment has been conducted in order to assess the accuracy of the proposed SHM network at reconstructing the natural oscillation frequencies and shapes of a simply supported steel beam. The discussed results reveal the potentialities of the implemented network for SHM applications thanks to its versatility and high scalability, becoming a suitable candidate for a relatively cheap and low consumption system capable to provide real-time information.

Data Availability

The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

Conflicts of Interest

The authors declare that there is no conflict of interests with any company or organization regarding the material discussed in this paper.

References

[1] J. P. Lynch and K. J. Loh, “A summary review of wireless sensors and sensor networks for structural health monitoring,” *The Shock and Vibration Digest*, vol. 38, no. 2, pp. 91–128, 2006.

[2] V. Pazzi, A. Lotti, P. Chiara, L. Lombardi, M. Nocentini, and N. Casagli, “Monitoring of the vibration induced on the arno masonry embankment wall by the conservation works after the May 25, 2016 riverbank landslide,” *Geoenvironmental Disasters*, vol. 4, no. 1, p. 6, 2017.

[3] J. Hester, S. Prabhu, S. Atamturktur, and J. Sorber, “Remote and wireless long-term vibration monitoring of historic monuments,” *Procedia Engineering*, vol. 199, pp. 3302–3307, 2017.

[4] G. Boscato, A. Dal Cin, S. Russo, and F. Sciarretta, “Shm of historic damaged churches,” *Advanced Materials Research*, vol. 838-841, pp. 2071–2078, 2013.

[5] A. Girolami, D. Brunelli, and L. Benini, “Low-cost and distributed health monitoring system for critical buildings,” in *2017 IEEE Workshop on Environmental, Energy, and Structural Monitoring Systems (EESMS)*, pp. 1–6, Milan, Italy, July 2017.

[6] J. Shinae and B. F. Spencer Jr, *Structural Health Monitoring for Bridge Structures Using Smart Sensors, Report No. nsel-035*, Technical report, Department of Civil and Environmental Engineering University of Illinois at Urbana-Champaign, 2015.

[7] Z. Chen, X. Zhou, X. Wang, L. Dong, and Y. Qian, “Deployment of a smart structural health monitoring system for long-span arch bridges: a review and a case study,” *Sensors*, vol. 17, no. 9, p. 2151, 2017.

[8] K. Chebrolu, B. Raman, N. Mishra, P. K. Valiveti, and R. Kumar, “Brimon: a sensor network system for railway bridge monitoring,” in *Proceeding of the 6th international conference on Mobile systems, applications, and services - MobiSys '08*, pp. 2–14, Breckenridge, CO, USA, June 2008.

[9] F. Moreu, R. E. Kim, and B. F. Spencer Jr, “Railroad bridge monitoring using wireless smart sensors,” *Structural Control and Health Monitoring*, vol. 24, no. 2, article e1863, 2017.

[10] X. Liu, S. Tang, and X. Xu, “Smart wireless sensor nodes for structural health monitoring,” *Intelligent Sensor Networks*, pp. 77–91, 2012.

[11] F. Federici, R. Alesii, A. Colarieti et al., “Design of wireless sensor nodes for structural health monitoring applications,” *Procedia Engineering*, vol. 87, pp. 1298–1301, 2014.

[12] S. Cho, C.-B. Yun, J. P. Lynch, A. T. Zimmerman, B. F. Spencer Jr, and T. Nagayama, “Smart wireless sensor technology for structural health monitoring of civil structures,” *Steel Structures*, vol. 8, no. 4, pp. 267–275, 2008.

[13] D.-W. Lim, S. C. Mantell, and P. J. Seiler, “Wireless monitoring algorithm for wind turbine blades using piezo-electric energy harvesters,” *Wind Energy*, vol. 20, no. 3, pp. 551–565, 2017.

[14] S. N. Pakzad, G. L. Fenves, S. Kim, and D. E. Culler, “Design and implementation of scalable wireless sensor network for structural monitoring,” *Journal of Infrastructure Systems*, vol. 14, no. 1, pp. 89–101, 2008.

[15] T. Torfs, T. Sterken, S. Brebels et al., “Low power wireless sensor network for building monitoring,” *IEEE Sensors Journal*, vol. 13, no. 3, pp. 909–915, 2013.

[16] G.-D. Zhou and T.-H. Yi, “Recent developments on wireless sensor networks technology for bridge health monitoring,” *Mathematical Problems in Engineering*, vol. 2013, 33 pages, 2013.

[17] M. Z. A. Bhuiyan, G. Wang, J. Cao, and J. Wu, “Deploying wireless sensor networks with fault-tolerance for structural health monitoring,” *IEEE Transactions on Computers*, vol. 64, no. 2, pp. 382–395, 2015.

[18] Z. A. B. Md, G. Wang, and K.-K. R. Choo, “Secured data collection for a cloud-enabled structural health monitoring system,” in *2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pp. 1226–1231, Sydney, NSW, Australia, December 2016.

[19] M. Z. Alam Bhuiyan, J. Wu, G. Wang, and J. Cao, “Sensing and decision making in cyber-physical systems: the case of structural event monitoring,” *IEEE Transactions on Industrial Informatics*, vol. 12, no. 6, pp. 2103–2114, 2016.

[20] A. Abdelgawad and K. Yelamarthi, “Internet of things (iot) platform for structure health monitoring,” *Wireless Communications and Mobile Computing*, vol. 2017, 10 pages, 2017.

[21] A. Abdelgawad and K. Yelamarthi, “Structural health monitoring: Internet of things application,” in *2016 IEEE 59th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pp. 1–4, October 2016.

[22] Y. Nakamura, Y. Arakawa, and K. Yasumoto, “Senstick: a rapid prototyping platform for sensorizing things,” in *2016 Ninth International Conference on Mobile Computing and Ubiquitous Networking (ICMU)*, pp. 1–6, Kaiserslautern, Germany, October 2016.

[23] P. Desai, A. Sheth, and P. Anantharam, “Semantic gateway as a service architecture for iot interoperability,” in *2015 IEEE International Conference on Mobile Services*, pp. 313–319, New York, NY, USA, June-July 2015.

[24] L. Console, I. Lombardi, C. Picardi, and R. Simeoni, “Toward a social web of intelligent things,” *AI Communications*, vol. 24, no. 3, pp. 265–279, 2011.

[25] T. Rabl, S. Gómez-Villamor, M. Sadoghi, V. Muntés-Mulero, H.-A. Jacobsen, and S. Mankovskii, “Solving big data challenges for enterprise application performance management,” *Proceedings of the VLDB Endowment*, vol. 5, no. 12, pp. 1724–1735, 2012.

[26] B. R. Chang, H.-F. Tsai, C.-Y. Chen, and C.-L. Guo, “Empirical analysis of high efficient remote cloud data center backup using hbase and cassandra,” *Scientific Programming*, vol. 2015, Article ID 294614, 10 pages, 2015.

[27] N. Testoni, L. De Marchi, A. Ferraro, and A. Marzani, “A coin size, 40mw, 20 grams sensor node for guided waves detection,” *Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems 2015*, 2015.

[28] W. Flores-Fuentes, M. Rivas-Lpez, D. Hernndez-Balbuena et al., “Online shm optical scanning data exchange,” in *2016 IEEE 25th International Symposium on Industrial Electronics (ISIE)*, pp. 940–945, Santa Clara, CA, USA, June 2016.

[29] R. P. Dramble, “Power consumption,” December, 2017 https://www.pidramble.com/wiki/benchmarks/power-consumption.

[30] T.-k. Chin and D. Tran, “Combine power feed and data link via cable for remote peripherals,” February, 2015 http://www.eetimes.com/document.asp?doc_id=1279219, 2011.

[31] A. Lakshman and P. Malik, “Cassandra: a decentralized structured storage system,” *ACM SIGOPS Operating Systems Review*, vol. 44, no. 2, pp. 35–40, 2010.

[32] P. Beynon-Davies, “Relational data model,” in *Database Systems*, pp. 23–41, Springer, 1996.

[33] STMicroelectronics, *Parameters and Calibration of a Low-g 3-Axis Accelerometer*, 2014.

[34] S.-h. P. Won and F. Golnaraghi, “A triaxial accelerometer calibration method using a mathematical model,” *IEEE Transactions on Instrumentation and Measurement*, vol. 59, no. 8, pp. 2144–2153, 2010.

[35] U. Qureshi and F. Golnaraghi, “An algorithm for the in-field calibration of a mems imu,” *IEEE Sensors Journal*, vol. 17, no. 22, pp. 7479–7486, 2017.

[36] M. Abe and I. I. I. Julius O Smith, “Design criteria for simple sinusoidal parameter estimation based on quadratic interpolation of fft magnitude peaks,” in *Audio Engineering Society Convention 117*, Audio Engineering Society, 2004.

[37] B. Peeters and G. De Roeck, “Stochastic system identification for operational modal analysis: a review,” *Journal of Dynamic Systems, Measurement, and Control*, vol. 123, no. 4, pp. 659–667, 2001.

[38] P. Stoica and R. L. Moses, *Spectral Analysis of Signals*, vol. 452, Pearson Prentice Hall, Upper Saddle River, NJ, 2005.

[39] T.-H. Le, Y. Tamura, A. Yoshida, and N. Dong, “Frequency domain versus time domain modal identifications for ambient excited,” *Structure*, vol. 6, 2010.

[40] V. Krishnamurthy, K. Fowler, and E. Sazonov, “The effect of time synchronization of wireless sensors on the modal analysis of structures,” *Smart Materials and Structures*, vol. 17, no. 5, article 055018, 2008.

[41] F. Poncelet, G. Kerschen, and J.-C. Golinval, *Experimental Modal Analysis Using Blind Source Separation Techniques*, International Conference on Noise and Vibration Engineering, Leuven, 2006.

[42] A. A. A. Donovan and B. W. Kernighan, *The Go Programming Language*, Addison-Wesley Professional, 2015.

[43] A. Chebotko, A. Kashlev, and S. Lu, “A big data modeling methodology for apache cassandra,” in *2015 IEEE International Congress on Big Data*, pp. 238–245, New York, NY, USA, June-July 2015.

[44] B. Swaminathan, B. Sharma, and S. Chauhan, “Utilization of blind source separation techniques for modal analysis,” in *Structural Dynamics*, T. Proulx, Ed., vol. 3 of Conference Proceedings of the Society for Experimental Mechanics Series, pp. 189–206, Springer, New York, NY, 2011.

[45] S. Park and N. Stubbs, “Reconstruction of mode shapes using Shannon’s sampling theorem and its application to the nondestructive damage localization algorithm,” in *Smart Structures and Materials 1995: Smart Systems for Bridges, Structures, and Highways*, vol. 2446, pp. 280–293, San Diego, CA, United States, April 1995.