A MODEL-BASED APPROACH FOR BRIDGE STRUCTURAL HEALTH MONITORING USING WIRELESS SENSOR NETWORKS

A Thesis

Submitted to the Graduate School of the University of Notre Dame in Partial Fulfillment of the Requirements for the Degree of Master of Science

by

Erez Abittan, Ingénieur Supélec

Panos J. Antsaklis, Director

Graduate Program in Electrical Engineering Notre Dame, Indiana

April 2006
A MODEL-BASED APPROACH FOR BRIDGE STRUCTURAL HEALTH MONITORING USING WIRELESS SENSOR NETWORKS

Abstract
by
Erez Abittan

A new method for bridge structural health monitoring using a wireless sensor network is introduced. This method uses heterogeneous sensors connected in a wireless sensor network and aims to detect damage in the structure at early stages while reducing the dependence of the method on exact knowledge of the excitation of the structure.

The wireless network adds many constraints in the algorithm used, and this work focuses on low energy consumption, low computational power and wireless sensors which may have unsynchronized clocks. The network architecture has two layers: the micro layer for local damage detection and the macro layer for data fusion.

At each node, two related measurements, acceleration and strain, are compared using appropriate models of the healthy bridge and residuals. If the measurements are different than expected, the structure may be damaged. At the higher level, all responses from the nodes are combined to detect whether the structure is healthy or not.

Novel contribution of this work include the problem formulation using heterogeneous sensors and a two-tiered network architecture, the detection scheme at each node using weighted thresholds and the data fusion at the higher level, and the associated simulations based on a finite element model of a cantilever beam. The goal has been to combine ideas from many fields in order to set the foundation for an approach in damage-detection using wireless sensors networks.
To my parents and my brothers
for their love and support.

A mes parents et mes frères
qui me manquent quand même quelques fois.
TABLE OF CONTENTS

LIST OF FIGURES ............................................................................................................. v

ACKNOWLEDGMENTS ........................................................................................................ vii

CHAPTER 1 BACKGROUND............................................................................................. 1
  1.1 Introduction ........................................................................................................ 1
  1.2 Structural health monitoring systems ............................................................. 2
  1.3 Process monitoring .......................................................................................... 3
  1.4 Wireless sensor networks ................................................................................ 4

CHAPTER 2 WIRELESS SENSOR NETWORKS FOR BRIDGE MONITORING .......... 7
  2.1 Problem presentation ..................................................................................... 7
  2.2 Related work ................................................................................................... 9
    2.2.1 Shifts in the modal frequencies and change in the mode shapes........ 10
    2.2.2 Change in the time series of structural response ................................. 11

CHAPTER 3 A NETWORK ARCHITECTURE............................................................... 15
  3.1 Decentralized two-tiered architecture ........................................................ 15
  3.2 Operational structure of the network and orchestration ............................... 17

CHAPTER 4 AN APPROACH TO LOCAL-BASED DAMAGE DETECTION .......... 19
  4.1 Model-based method for fault detection ...................................................... 19
    4.1.1 Operational condition for monitoring .............................................. 19
    4.1.2 Unknown input model based fault detection .................................... 20
    4.1.3 Analytical demonstration ................................................................. 23
  4.2 Algorithm to be embedded ......................................................................... 25
    4.2.1 Lower tier architecture ................................................................. 25
    4.2.2 Practical residual generation ......................................................... 26
    4.2.3 Local decision making ................................................................. 27
    4.2.4 Performance of the processing ..................................................... 28
  4.3 Simulations ..................................................................................................... 29
    4.3.1 Experiment set up .......................................................................... 29
    4.3.2 Implementation of the model based approach ................................. 33
    4.3.3 Detection performances for the model-based approach .................... 36
### LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Decentralized network architecture</td>
<td>15</td>
</tr>
<tr>
<td>2</td>
<td>Two-tiered architecture</td>
<td>16</td>
</tr>
<tr>
<td>3</td>
<td>Information processing at each level</td>
<td>16</td>
</tr>
<tr>
<td>4</td>
<td>Sequencing of tasks in the network</td>
<td>18</td>
</tr>
<tr>
<td>5</td>
<td>Decentralized model-based process monitoring</td>
<td>21</td>
</tr>
<tr>
<td>6</td>
<td>Single-Input Multi-Outputs system</td>
<td>21</td>
</tr>
<tr>
<td>7</td>
<td>Model-Based Structural Health Monitoring system</td>
<td>22</td>
</tr>
<tr>
<td>8</td>
<td>Lower tier architecture</td>
<td>25</td>
</tr>
<tr>
<td>9</td>
<td>Residual generation</td>
<td>26</td>
</tr>
<tr>
<td>10</td>
<td>Loading scenario</td>
<td>29</td>
</tr>
<tr>
<td>11</td>
<td>Geometrical characteristics of the beam modeled</td>
<td>30</td>
</tr>
<tr>
<td>12</td>
<td>Changes in the time series signals for a damaged beam</td>
<td>31</td>
</tr>
<tr>
<td>13</td>
<td>Characteristics of the input signal</td>
<td>32</td>
</tr>
<tr>
<td>14</td>
<td>Modeling errors</td>
<td>34</td>
</tr>
<tr>
<td>15</td>
<td>Residual number divided by the dynamic threshold for alpha = 1</td>
<td>35</td>
</tr>
<tr>
<td>16</td>
<td>Determination of the scaling parameter alpha</td>
<td>36</td>
</tr>
<tr>
<td>17</td>
<td>Fault detection with the model-based approach and no noise</td>
<td>37</td>
</tr>
<tr>
<td>18</td>
<td>Determination of the scaling parameter alpha for a noisy environment</td>
<td>38</td>
</tr>
<tr>
<td>19</td>
<td>Fault detection with the model-based approach and noise with SNR=40dB</td>
<td>39</td>
</tr>
<tr>
<td>20</td>
<td>Fault detection with the model-based approach and noise with SNR=20dB</td>
<td>40</td>
</tr>
<tr>
<td>21</td>
<td>Fault detection with the LANL method and no noise</td>
<td>41</td>
</tr>
</tbody>
</table>
Figure 22 Fault detection with the LANL method and noise with SNR=40dB ............42
Figure 23 Fault detection with the LANL method and noise with SNR=20dB ............42
Figure 24 Model-based approach with input signals inside of the training set ..........44
Figure 25 Model-based approach with inputs’ duration outside of the training set ....44
Figure 26 Model-based approach with inputs’ amplitude outside of the training set .....45
Figure 27 Bell-shaped input signal ........................................................................46
Figure 28 Model-based approach with bell-shaped inputs signal on the structure ......46
Figure 29 Decision and data fusion using a voting scheme ........................................51
Figure 30 Global decision making .........................................................................52
Figure 31 Ideal decision making at the higher tier ......................................................53
Figure 32 Influence of the number of sensors used in the voting scheme ...............55
Figure 33 Location of smart sensors on the beam ......................................................56
Figure 34 Data fusion using voting scheme for three smart sensors .......................57
Figure 35 Data fusion using the voting scheme for three smart sensors overtime ....58
Figure 36 Location of the sensors .............................................................................61
ACKNOWLEDGMENTS

First, I would like to express my sincere gratitude to my advisor, Professor Panos J. Antsaklis for his advice, encouragement and support during my two years of study at the University of Notre Dame.

I wish to acknowledge the help of Mr. Su Su, graduate student from the Civil Engineering department at the University of Notre Dame. He developed the finite element model used to generate strain and acceleration signals in the Matlab simulations of a cantilever beam (§ 4.3). I would also like to acknowledge the many useful discussions with Professor Tracy Kijewski-Correa, Professor of Civil Engineering at the University of Notre Dame.

Finally, I would like to thank the readers on my committee: Professor Panos J. Antsaklis, Professor Martin Haenggi and Professor Tracy Kijewski-Correa for their time and their constructive comments.
CHAPTER 1

BACKGROUND

1.1 Introduction

The area of Structural Health Monitoring (SHM) is a candidate for revolutionary improvement with the introduction of the Wireless Sensor Networks (WSN) technology. The current research in Structural Health Monitoring aims to develop systems that could identify a fault, localize the damage area and even estimate the remaining life time of the structure. This requires an increasing number of sensors, and the wiring cost for a big structure becomes a real issue.

On the other hand, maintenance costs of bridges in remote areas are huge as each bridge should be inspected every 6 months to insure users’ security. However, funds communities can spend for this kind of maintenance are not adequate; for example in the state of Indiana bridges are typically inspected only every two years at best. This proves the existence of a market for a low cost and reliable system that would be able to perform a first level analysis of the structure’s health and prevent users from risky transits over bridges.

Such an approach to Structural Health Monitoring of bridges is developed here. The motivation is to find a solution to the need expressed by communities owning bridges in remote areas; it is about designing a low cost and reliable system that would monitor the structure’s health and detect damage. This information could then be used, if
a fault is detected, to request the dispatch of a team who would perform further inspections.

This problem is at the intersection of three main areas; Process monitoring, Structural Health Monitoring and Wireless Sensor Networks. First, it is necessary to develop a background in these fields; then it becomes possible to explore the Wireless Sensors Networks application in Structural Health Monitoring and the model-based approach introduced in this report.

1.2 Structural health monitoring systems

Structures are exposed to external conditions that are susceptible to change drastically from mild temperature and little mechanical solicitation to extreme conditions including heavy snow, or hurricanes, and earthquake. In order to monitor these structures and make sure that users are not taking any risk, the first type of system developed has been called structural monitoring system. The goal is to record structural signals like stress, strain or acceleration when the structure is subject to excitation and to store the data.

In the recent years, the trend has been to design systems called structural health monitoring systems which are able to monitor the structure and in addition detect damage automatically. This new kind of system requires the development of an automated method able to perform quantitative damage detection on the global structure with an emphasis on detecting damage at its early age to decrease the repair costs. The literature available in the field of detection methods is very broad [1, 2] and we will provide the reader with an overview of the different trends in this area. These methods can be classified as either local based and global based.

Local based damage detection methods identify damage by working on subsets of the whole structure monitored. This means that the possible damage location needs to be known or the whole structure needs to be screened using local based damage
detection methods. These methods include visual inspection by an operator, local vibration signals analysis, acoustic or ultrasonic methods, magnetic field analysis, radiography, eddy current methods or thermal field methods.

On the other hand, global based damage detection methods tend to consider the global vibration characteristics of the structure. These methods are working mainly by looking for changes in the modal parameters, i.e. mode shape and natural frequencies.

Many techniques exist and many systems have been designed to detect damage in a structure. The next step in Structural Health Monitoring is the design of more autonomous, reliable and cheaper systems for massive implementation in structures around the world.

1.3 Process monitoring

Model-based fault detection in the process monitoring area has been a broad and dynamic field for more than thirty years [3]. Concepts of fault detection and localization are part of this field and it is not a surprise if some methods similar to those used to monitor structures are also used to monitor processes. When using a model for a higher level of abstraction, the problems of plant monitoring and structural monitoring require similar solutions. A review of the main trends in process monitoring is presented as background to take advantage of the research done in this area and provide the reader with another point of view on fault-detection systems.

The most common way to monitor a process is to compare the actual behavior of the monitored plant to the expected behavior predicted via a model. In the case of systems monitored by measuring vibrations signals, the idea is to use observations – measurements of vibrations signals – to generate a residual vector which represents how far the actual behavior is from the usual condition. It is possible to make a classification in two families of the methods used to generate a residual: those relying only on data processing and those using some additional knowledge about the process [4].
Data driven methods are those using only measurements from the system and no specific prior knowledge about what is the process’ behavior. Most techniques (Principal Component Analysis, Fisher Discriminant Analysis or Partial Least Square Method) are based on dimensionality reduction. The idea is to project the observed data onto a set of vectors characterizing the measured data of the non faulty process. In the case of a fault in the process, this makes it possible to observe variation in the observation space which will appear.

Another common way to perform process monitoring is to use analytical and knowledge-based methods. Based on the measurement of the inputs and outputs of a process, analytical methods generate features using detailed mathematical models— the most popular being residual signals – and then compare these features to those associated with the non-faulty process. Methods generating residuals are referred as analytical redundancy and use three main techniques: parameter estimation, observers and parity relation. In all cases the idea is to compare the measured outputs with the ones estimated using the input and a model of the process.

The area of process monitoring has been very active for many years and the methods developed prove their capacity to detect a fault at its early age as they have been used in factories all around the world. It might be in the interest of researchers from the structural health monitoring area to import some of this experience in their field; this would bring new ideas and maybe help them design the structural health monitoring system of the future.

1.4 Wireless sensor networks

With the recent technology improvements that reduced size and/or cost of MEMS (Micro Electro-Mechanical System) sensors, microcontrollers, wireless communication equipment and circuit design, a new kind of embedded systems called wireless sensor networks emerged. Such systems are networks of embedded devices that
combine a sensing unit and/or actuators, some limited computational capabilities some memory and a radio to communicate with other devices in its neighborhood [5]. The vision associated with this new technology is to see the size and the cost of these systems decreasing down to the point where this technology would disappear in our everyday life and become present everywhere.

Even if it is not obvious because of the name “Wireless Sensor Networks”, it is important to remember that these networks are made of “smart sensors” able to add value to the sensed signal by processing and collaboration with other devices. Single devices from the network are also called “sensor node”, node, or even “mote” as a reference to the miniaturization.

Miniaturization is not the only advantage of the wireless sensor networks technology. We should also point out (a) the wireless communication unit that makes network implementation much easier and makes possible a network of tiny mobile agents and increases the flexibility of the network structure, (b) the low cost target of such a device and (c) the possibility to process information in a decentralized manner. With these characteristics wireless sensor networks add more value than other technologies: they create networked systems that are scalable (because of the possibility to increase the number of devices in the network more easily) and robust (if a node of the network fails, the remaining nodes can still perform the global task).

On the other hand, these advantages, at the same time, are adding some more constraints to the system. First, if the network uses wireless communication, it is to increase flexibility so most of the times, it is not possible to have a wired connection to a power supply: each device uses its own embedded and limited power supply. In addition, as motes might be embedded in places out of reach, inside a structure for example, replacing the autonomous power supply can be a major problem and saving energy becomes a priority. Also, it is not achievable to synchronize clocks of all devices in the network and most of the ways to improve synchronization increase the energy
consumption. Last, because of the limited computational capabilities, it is necessary to use algorithms as simple as possible.

Then when designing a system relying on the wireless sensor networks technology, the main challenge is to use the full potential of the technology (decentralized processing, scalability…) while dealing with the induced limitation of such networks.
CHAPTER 2
WIRELESS SENSOR NETWORKS
FOR BRIDGE MONITORING

2.1 Problem presentation

As the trend in Structural Health Monitoring (SHM) is to design “intelligent” systems that can automatically detect damage in a structure, there is a need for an increased number of sensors in order to analyze responses both locally and globally. For example, the Wind and Structural Health Monitoring system used by the Hong Kong Highway Department requires approximately 900 sensors and costs $1.3 million to monitor the structural behavior of three bridges that run between Hong Kong and its airport. The wiring, deployment and maintenance cost of a large number of sensors becomes an issue which leads to the use of a Wireless Sensor Network. Moreover, as there is a need for a robust, simple and low cost system, a Wireless Sensor Network maybe the way to achieve this goal – the traditional centralized network architecture implemented with wireless devices does not provide adequate robustness – and it is not only a way to cancel the wiring costs.

Even if the wireless sensor networks technology seems to be the best choice to implement the new generation of Structural Health Monitoring systems; it will raise some constraints that need to be addressed during the design phase. We will focus mainly on the following limitations:

> Energy constraint: In order to take full advantage of the wireless capabilities, we need to get rid of the power cable. As a lot of research has been done in
this area, in the future, it might be possible to think about self powered motes [6] or about powerless motes using the RFID technology [7]. However, as there is no viable solution available yet, we need to limit the network to be used only “on demand” and for a limited time to increase its life time.

> Limited number of messages exchanged: As most of the energy consumption in a mote comes from the communication, there is a need to limit the number of messages exchanged. In other words, the decentralized architecture of the network should rely on as few message exchanges as possible. At the high level, one way of doing that would be to emphasize local processing to communicate with neighbors only when there is no other possibility.

> Limited computational power: In order to lower the energy consumption, and to lower the hardware cost, most of the motes available in the market have an 8-bit microprocessor. Therefore, we need to be aware of the algorithm complexity so that computation time and/or energy consumption does not increase too much relative to the number of samples considered.

> Difficulties to synchronize clocks: The decentralized structure of the network often relies on a centralized resource: a common clock. There are ways to perform a clock synchronization (Time averaging, Master election, Broadcast [8]) but most of them require a fair number of packet exchanges between motes – the larger the number of packets sent, the better the synchronization. Then, in order to increase the power efficiency of the network, we would like to relax the “synchronized network” assumption.

In addition, the structure monitored, a bridge for example, will introduce some additional constraints due to a wide variability of its behavior. The response of the structure caused by a specified input – i.e. due to the same car traffic for example – may depend on its temperature, the humidity around, the wind, even on the moon position. These external factors that modify the bridge behavior can be structured in two sets: the ones that occur periodically – due to the season and time, summer hot and humid at noon and winter very cold at night for instance – and the ones that vary instantaneously – such as a storm or a windy day. Thus we can have some idea on how the bridge will behave at a certain time. It is also possible to use additional sensors to gain information about the external conditions that will affect the structure’s behavior and have a more accurate estimation of what behavior to expect.
The intelligent damage detection problem can be broken down into five subproblems in a hierarchical manner. From the broader to the more detailed, according to the work following Anders Rytter [9], we have:

1. Detection: Qualitative indication that damage might be present in the structure.
2. Localization: Information about probable position of the damage
3. Classification (Optional step): Information about the type of damage
4. Assessment: Estimate of the extent of the damage
5. Prediction: Estimate of residual life

The first step for SHM using a Wireless Sensor Network is to solve the detection problem. Then in a decentralized architecture, with the assumption that a sensor closer to the damage will detect a more significant change, localization the damage might be possible. The idea would be to identify which sensor or set of sensors were the most able to detect the damage and assume that it is localized close to these sensors.

Research on wireless sensor networks to perform structural health monitoring is at its early stage and there are still plenty of challenges that need to be addressed. Here, we are interested in solving the problem of fault detection using a low-cost and reliable system. In case of fault detection, it is then possible to send an operator on site to investigate the remaining steps in the damage detection process as described in [9].

2.2 Related work

Most research has been done in the acquisition and compression of data through a wireless sensor network, with processing at a centralized unit. This idea is to emulate the structure of traditional SHM systems using wireless links like the multi-hop wireless data acquisition system for structural health monitoring WISDEN [10] developed by a USC group. It shows great results but it is not adequate for using it in a remote area and it does not use the full potential of a sensor network: the decentralized computation capabilities.

The damage detection problem is solved mainly by measuring and analyzing structural vibrations. This response is composed of several harmonics and each
harmonic, called mode, is defined by its frequency and its shape which are sensitive to the health of the structure. Then structural health monitoring is performed by looking for changes in the modal properties of the structure. In the implementation using a wireless sensor networks, the more popular methods are the study of shifts in the modal frequencies and the study of changes in the mode shape and study of changes in the time series of structural response [11].

2.2.1 Shifts in the modal frequencies and change in the mode shapes

This method is one of the first used for damage detection using a wireless sensor networks. As the frequency response – set of the modes – has a strong connection with the physics of the structure, a change in the modes (shape or frequency) is related to a change in the physics, i.e., to a potential damage.

This idea is more intuitive for one who has a basic background in structures. In traditional SHM systems, these methods have been used [12] and provided good results especially for high levels of damage [13]. Nevertheless, there are two main drawbacks with these methods when implemented in a decentralized manner with a wireless sensor network.

First, this analysis occurs in the frequency domain which requires computing the Fourier Transform of a discrete sequence. For $N$ samples, the order of the complexity is $O(N^2)$ to compute the Discrete Fourier Transform and $O(N \log_2(N))$ to compute the Discrete Fast Fourier Transform. This can be energy consuming for the type of platforms considered for the embedment, especially if we consider keeping the wireless sensor network working for a long life time.

Furthermore, it is very tricky to deal with modes because of their variability in space and in the operational condition. At some locations, two modes can cancel each other, or a mode can be hidden behind some other mode. Also modes are very sensitive
to external conditions and a change in the temperature can shift a mode or change its shape while the health of the structure remains the same.

2.2.2 Change in the time series of structural response

This method, developed in the Los Alamos National Laboratory [14] and implemented later by Lynch et al. [12], is based on the idea of statistical pattern recognition. It uses a two stage prediction model with first an auto-regressive model (AR) and then an auto-regressive model with exogenous inputs (ARX). At the current stage of research in the implementation of a structural health monitoring algorithm using a wireless sensor networks, this technique showed the best results. Then, in order for one to understand the relevance of the approach introduced in this study, it is necessary to get more familiar with the LANL technique, which works in two steps.

First, there is the set up of the system: a data base is setup to keep information about vibration measurement for the healthy structure under any kind of operational condition (temperature, humidity, wind...). This step requires for each operational condition $i$ the following actions: All vibration signals $x_i$, of length $N_i$, are standardized:

$$
\hat{x}_i = \frac{x_i - \mu_{x_i}}{\sigma_{x_i}} \quad \text{with} \quad \left\{ \begin{array}{l} 
\sigma_{x_i} = \sqrt{\text{var}(x_i)} \\
\mu_{x_i} = \text{mean}(x_i) 
\end{array} \right.
$$

(2.1)

Then an AR model of order $p$ (with typically $p = 30$ to accommodate noise and other disturbances) is constructed to fit the data, and the coefficients $\phi_{x_i}$ are stored in a data base:

$$
\hat{x}_i(t) = \sum_{j=1}^{p} \phi_{x_i,j} \cdot \hat{x}_i(t-j) + e_{x_i}(t), \quad \text{for} \quad t = 1, 2, \ldots, N_x
$$

(2.2)

where $e_{x_i}(t)$ is the error signal associated with the AR model used.
Last, based on the assumption that the error $e_{yi}(t)$ is mainly caused by the unknown external input an ARX model ARX($a,b$) with $a + b \leq p$, typically $a = b = 5$, is used to reconstruct the input/output relationship between $e_{yi}(t)$ and $\hat{x}_i(t)$:

$$\hat{x}_i(t) = \sum_{m=1}^{a} \alpha_{i,m} \cdot \hat{x}_i(t-m) + \sum_{n=1}^{b} \beta_{i,n} \cdot e_{yi}(t-n) + e_{yi}(t)$$  \hspace{1cm} (2.3)

Where $e_{yi}(t)$ is the error signal associated with the ARX model used.

There is not a unique way to determine the coefficient of the ARX model in equation (2.3) as the solution depends on the criterion of fit used to solve this optimization problem. The least square method is used here [15] because it is the simplest way to determine the ARX coefficients:

$$\begin{bmatrix} \alpha_{i,1} \\ \vdots \\ \alpha_{i,a} \\ \beta_{i,1} \\ \vdots \\ \beta_{i,b} \end{bmatrix} = \left( \begin{bmatrix} \hat{x}_i(t-1) \\ \vdots \\ \hat{x}_i(t-a) \\ e_{yi}(t-1) \\ \vdots \\ e_{yi}(t-b) \end{bmatrix} \right)^{-1} \cdot \left( \begin{bmatrix} \hat{x}_i(t-1) \\ \vdots \\ \hat{x}_i(t-a) \\ e_{yi}(t-1) \\ \vdots \\ e_{yi}(t-b) \end{bmatrix} \right)$$  \hspace{1cm} (2.4)

To perform damage detection, $\alpha_{i,m}, \beta_{i,n}$ and $\sigma^2(\epsilon_{yi}) = E[(\epsilon_{yi} - \mu_{\epsilon_{yi}})^2]$ are stored in the database. Therefore, we have $\{\phi_1, \phi_2, \ldots, \phi_p\}, \alpha_{i,m}, \beta_{i,n}$ and $\sigma^2(\epsilon_{yi})$ stored in the database for each operational condition $i$.

Second, after the previous set up, the system can be used for health monitoring. When an analysis of the structure’s health is performed, the measured vibration signals $y_i$ are processed in the following way. Time signals of length $N_y$ are standardized:

$$\hat{y}_i = \frac{y_i - \mu_{y_i}}{\sigma_{y_i}}$$ \hspace{1cm} (2.5)
And an AR model is computed using the Yule-Walker equations to fit the data so that:

$$\hat{y}(t) = \sum_{j=1}^{p} \phi_{y_j} \cdot \hat{y}(t-j) + e_y(t)$$

(2.6)

Using

$$
\begin{bmatrix}
R_{yy}(0) & R_{yy}(1) & \ldots & R_{yy}(p-1) \\
R_{yy}(1) & R_{yy}(0) & \ldots & R_{yy}(p-2) \\
\vdots & \vdots & \ddots & \vdots \\
R_{yy}(p-1) & R_{yy}(p-2) & \ldots & R_{yy}(0)
\end{bmatrix}
\begin{bmatrix}
\phi_{1} \\
\phi_{2} \\
\vdots \\
\phi_{p}
\end{bmatrix}
= 
\begin{bmatrix}
R_{yy}(1) \\
R_{yy}(2) \\
\vdots \\
R_{yy}(p)
\end{bmatrix}
$$

(2.7)

Where \( R_{yy}(k) \) is the autocorrelation function of \( \hat{y}_i \) for \( k \) points of lag:

$$R_{yy}(k) = \frac{1}{N_y-k} \cdot \sum_{n=1}^{N_y-k} \hat{y}(n) \cdot \hat{y}(n+k)$$

(2.8)

This step requires \( p \cdot (N-p) \) additions and multiplication for computing the coefficient of the matrix and then it is need to invert a matrix of dimension \( p^2 \) to solve the system of equations.

Then a search in the database looks for the closest \( \{\phi_{1},\phi_{2},\ldots,\phi_{p}\} \) coefficients to the ones computed from the sequence \( \hat{y} \). The goal is to identify which operational condition \( i \) from those recorded in the database for the healthy structure may correspond to the actual operational condition. Then, the corresponding \( \alpha_{i,m}, \beta_{i,n} \) and \( \sigma^2(e_{y_i}) \) are transmitted to the mote.

Next, to verify whether the actual structural condition is healthy or not, the difference between the measurement and the ARX model associated with the operational condition \( i \) is observed, i.e.:

$$e_{\hat{y}_i}(t) = \hat{y}(t) - \sum_{m=1}^{a} \alpha_{i,m} \cdot \hat{y}(t-m) + \sum_{n=1}^{b} \beta_{i,n} \cdot e_{\hat{y}}(t-n)$$

(2.9)

Last, the decision making is performed by comparing \( \sigma^2(e_{\hat{y}_i}) \) and \( \sigma^2(e_{x_i}) \). The idea underlying the test is that if the structure is in a healthy stage working under the same operational condition as the one used as reference, these two signals should have similar
statistical properties. According to the work of Sohn and Farrar [14] and under the assumptions of a normal distribution for $\varepsilon_{\hat{y}}$ and $\varepsilon_{\hat{x}}$, the test can be expressed as:

$$\frac{\sigma^2(\varepsilon_{\hat{y}})}{\sigma^2(\varepsilon_{\hat{x}})} > F_{n_y-1,n_x-1}^{\alpha} = h$$

(2.10)

Where $F_{n_y-1,n_x-1}^{\alpha}$ represent the upper 100·$\alpha$ percentile of an $F$-distribution with $N_y - 1$ and $N_x - 1$ degrees of freedom – $N_y$ and $N_x$ are the number of sample of signals $y$ and $x$.

According to the literature, this method has some good performance to identify and localize fault in a structure. However, it requires a lot of message exchanges, some processing power to compute the AR coefficient at each mote as well as a search in a database which makes this approach too energy and memory consuming for being embedded in a device for a life time use. This technique has been implemented using a wireless sensor network in a centralized [16] and decentralized [17] way, but in both cases, there was a need for a central database as a single mote does not carry enough memory to store all the pool data for the undamaged structure. It is, then, not compatible with a reliable and power efficient architecture where each mote is almost totally independent from the rest of the network (for reliability in case of failure and for limiting the power consuming communication with the neighborhood).

Most of the related approaches already proved their capability to detect and sometimes even localize damage in structures. However, up to date there is no fully decentralized system developed by academic researchers or industrial practitioners able to keep the network alive for an extended period of time. One can try to reduce the complexity of the algorithm to fit these requirements, and this is the basis of the approach developed in the following sections.
3.1 Decentralized two-tiered architecture

In order to provide a great level of robustness even with the use of low cost devices, it is necessary to use the full capabilities of wireless sensor networks and consider a decentralized architecture for the network.

The idea of a two-tier architecture has been used many times [18,19] because it has the capability to reduce energy consumption and monitor the structure on local and global levels at the same time. The suggestion here is a mixed architecture, with wired links on the lower tier to insure time synchronization of measurements from one neighborhood and wireless links on the higher level to suppress wiring cost and facilitate deployment. With this architecture, there is no need for a centralized clock as processing and comparison of related measurements are done locally under the same clock. The
requirement becomes an event-synchronization – less constrained than time synchronization as it does not require a common clock– to compare similar events when performing data fusion. The structure is monitored locally with the lower tier; then the decision for the global structure is made with the higher tier. As shown on the figure below, the lower tier is a set of sensors connected with a wire to the higher tier, which is the set of motes connected wirelessly.

![Two-tiered architecture diagram](image)

**Figure 2 Two-tiered architecture**

Information is processed at each level so that the network will bridge the gap between the bridge’s behavior and the user awareness of a fault.

![Information processing at each level diagram](image)

**Figure 3 Information processing at each level**
3.2 Operational structure of the network and orchestration

In order to maximize the sleeping time of each mote – it is the best way to expand the life time of the network – the network will have some lookout motes at the boundary to monitor the traffic crossing the bridge. When the input excitation and the operational condition are adequate, the lookup motes will wake up the whole network to perform an analysis of the structure.

When the lookup mote sends an alert, the different tasks are accomplished in the following order:

1. The lookup mote wakes up the network.
2. Each mote waits for the best measurements to perform local damage detection.
3. After making sure each mote is done with the previous tasks; the higher tier fuses local damage detection reports to create the global report on the structure.
4. Then the network communicates its results immediately if damage is detected or periodically otherwise either using an Internet connection (with many access points in case of mote failure) or by communicating to a portable device in the neighborhood (palm pilot, laptop, cell phone …).

This sequence is illustrated in the following flow chart.
Figure 4 Sequencing of tasks in the network
CHAPTER 4
AN APPROACH TO LOCAL-BASED
DAMAGE DETECTION

4.1 Model-based method for fault detection.

Traditional fault detection methods require the input of the system to be applied to the model, and then they compare the actual behavior to the one obtained by the model. For the problem of bridge monitoring, we assume that we don’t have access to the system’s input. To solve this problem, we combine two techniques: First, when analysis of the structure is performed, we try to narrow the set of possible inputs exciting the bridge by using the network only under similar operational conditions. Then we introduce a model based method based on two sets of measurements, one used as “input” and the other one as “output” so that the knowledge of the actual input of the whole system is not necessary.

4.1.1 Operational condition for monitoring

The common input used to perform system identification is a step as it is easily repeatable and creates, in the state of the system, a dynamic change that injects enough energy into the system to get measurements with a good signal-to-noise ratio. As it is currently not possible to use an actuator because of the difficulty to implement and use it in the real world [20], we can consider using vehicle traffic on the bridge as an input. More specifically the ideal would be a heavy truck crossing the bridge alone, each time with the same speed and under the same external condition. This will never happen and
an approximation would be to let the network be asleep and have some sensors at the boundary to wait for a high energy input. If the acceleration signal in the lookout mote is greater than a threshold, the mote will wake up the whole network to perform data acquisition. It remains necessary to deal with an unknown input but the range of possible operating condition is greatly reduced.

4.1.2 Unknown input model based fault detection

As energy consumption is one of the main constraints introduced by the use of a wireless sensor network, there is a need for an algorithm using “few” computations. The model based approach used in process monitoring has some advantages regarding this point: if the model is linear, there is no need to deal with solving non linear equations. In addition, if it possible to model the system using polynomials of reasonable order, there is no need to deal with big size matrices and all computation can be done in the time domain.

Nevertheless, the regular model based approach used in process monitoring requires a centralized architecture and needs to be adapted to the wireless sensor network environment. Instead of a model based monitoring of the global system, the system is virtually broken down into subsets monitored independently. The next step is to learn about the health of the global system by using data fusion.
It is very difficult, if not impossible, to measure the actual input signal either for the global system or for a subset of it. The proposed solution is, instead of considering the actual input, to consider part of the measured outputs as an “input”. This approach is derived from the representation of Single-Input Multi-Outputs systems using transfer function matrices. This can be illustrated by the following system:

**Figure 5 Decentralized model-based process monitoring**

**Figure 6 Single-Input Multi-Outputs system**
Where $U$ is the input, $Y_1$ and $Y_2$, such that $Y = \begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix}$, are the outputs and the transfer function matrix for this system is $G = \begin{bmatrix} G_1 \\ G_2 \end{bmatrix}$ so there is the relation:

$$Y = G \cdot U \quad \text{i.e.} \quad \begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix} = \begin{bmatrix} G_1 \\ G_2 \end{bmatrix} \cdot U$$

(4.1)

where $Y, G \in \mathbb{R}^{2 \times 1}$

In the case where $G_1^{-1}$ exists, we can have:

$$Y_2 = G_2 \cdot G_1^{-1} \cdot Y_1$$

(4.2)

i.e. $Y_2 = H_{21} \cdot Y_1$ with $H_{21} = G_2 \cdot G_1^{-1}$

(4.3)

Then for each subset of the bridge, the model-based approach for structural health monitoring using two measurements can be represented by the block diagram shown in Figure 7.

![Model-Based Structural Health Monitoring system](image)

**Figure 7 Model-Based Structural Health Monitoring system**

Here the demonstration uses a two outputs system but it can be generalized to an $n$ outputs system by simply splitting the $n$ outputs into two sets:

$$S_1 = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_i \end{bmatrix} \quad \text{and} \quad S_2 = \begin{bmatrix} Y_{i+1} \\ Y_{i+2} \\ \vdots \\ Y_n \end{bmatrix}$$

(4.4)
This can, under assumptions regarding the transfer function matrix $G$ similar to those discussed previously, lead to $S_2 = T_{21} \cdot S_1$ and then a comparable framework such as the one introduced here can be used.

This method works under the assumption that a change in the bridge’s parameters will change the relation between the two sets of measurements. It is possible to show analytically why this assumption makes sense and how this method can detect damage in the system.

### 4.1.3 Analytical demonstration

Considering the system $S$ with transfer function matrix $G$ input $U$ and outputs $Y_1$ and $Y_2$:

$$Y = G \cdot U$$  \hspace{1cm} (4.5)

We will use a model of the system $\hat{G}$ and as there are some modeling errors:

$$\hat{G} \approx G$$  \hspace{1cm} (4.6)

i.e. $G = \hat{G} + \Delta G$ ; where $\Delta G = \begin{bmatrix} \Delta G_1 \\ \Delta G_2 \end{bmatrix}$

Then under normal conditions, the following relation is satisfied:

$$\hat{Y} = \hat{G} \cdot U \approx Y$$  \hspace{1cm} (4.7)

And we can define an estimation of the output $Y_2$ using the model and its relation with $Y_1$

$$\tilde{Y}_2 = \hat{G}_2 \cdot \hat{G}_1^{-1} \cdot Y_1$$  \hspace{1cm} (4.8)

i.e. $\tilde{Y}_2 = \tilde{H}_{21} \cdot Y_1 \approx Y_2$  \hspace{1cm} (4.9)

When there is a fault in the system, its behavior will change. A fault in the structure is equivalent to a change in the system’s parameters which can be modeled by terms added to the transfer function. The system behavior follows this equation:

$$Y = (G + F) \cdot U$$  \hspace{1cm} (4.10)
i.e. \[
\begin{bmatrix}
Y_1 \\
Y_2
\end{bmatrix} = 
\begin{bmatrix}
G_1 + F_1 \\
G_2 + F_2
\end{bmatrix} \cdot U
\] (4.11)

Where \[\mathbf{F} = \begin{bmatrix} F_1 \\ F_2 \end{bmatrix}\] and \(\mathbf{F} = 0\) if and only if there is no fault in the system.

Then, in the case where \((G_1 + F_1)^{-1}\) exists, mathematically, we have:

\[Y_2 = (G_2 + F_2) \cdot (G_1 + F_1)^{-1} \cdot Y_1\] (4.12)

And the expression for the residual becomes:

\[r = Y_2 - \hat{Y}_2\] (4.13)

\[r = (G_2 + F_2) \cdot (G_1 + F_1)^{-1} \cdot Y_1 - \hat{G}_2 \cdot \hat{G}_1^{-1} \cdot Y_1\]

\[r = \left((G_2 + F_2) \cdot (G_1 + F_1)^{-1} - \hat{G}_2 \cdot \hat{G}_1^{-1}\right) \cdot Y_1\]

\[r = \left((G_2 + F_2) \cdot G_1^{-1} \cdot (1 + G_1^{-1} \cdot F_1)^{-1} - \hat{G}_2 \cdot \hat{G}_1^{-1}\right) \cdot Y_1\] (4.14)

And \(G_1^{-1} \cdot F_1 << 1\) because the fault \(F_1\) is considered at its early stage

\[r \approx \left((G_2 + F_2) \cdot G_1^{-1} \cdot (1 - G_1^{-1} \cdot F_1) - \hat{G}_2 \cdot \hat{G}_1^{-1}\right) \cdot Y_1\] (4.15)

i.e. \[r \approx (G_2 \cdot G_1^{-1} - \hat{G}_2 \cdot \hat{G}_1^{-1}) \cdot Y_1 + \left(F_2 \cdot G_1^{-1} \cdot (1 - G_1^{-1} \cdot F_1) - G_2 \cdot G_1^{-2} \cdot F_1\right) \cdot Y_1\] (4.16)

\[
\begin{aligned}
\text{\(r \approx (G_2 \cdot G_1^{-1} - \hat{G}_2 \cdot \hat{G}_1^{-1}) \cdot Y_1 + (F_2 \cdot G_1^{-1} \cdot (1 - G_1^{-1} \cdot F_1) - G_2 \cdot G_1^{-2} \cdot F_1) \cdot Y_1\)}
\end{aligned}
\]

\[
\begin{aligned}
\rightarrow 0 & \quad \text{Corresponds to modeling errors} \\
\text{Fault-sensitive part}
\end{aligned}
\]

Making the assumption \(G_1 \approx \hat{G}_1\) and \(G_2 \approx \hat{G}_2\) i.e. \(\Delta G_1 \rightarrow 0\) and \(\Delta G_2 \rightarrow 0\), which is reasonable as it is possible to adapt the model to make it fit the experimental data, this analysis shows that the residual \(r\) is null if and only if \[\begin{bmatrix}
F_1 \\
F_2
\end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}\]

Two important remarks should be made: First, the form of the fault sensitive part of the residual equation depends on the “input” \(Y_i\). When a fault occurs in the system, the amplitude of the residual signal will also depend on \(Y_i\). This point needs to be
addressed when designing the “Decision making”. Second, the behavior of the structure, and subsequently the matrix $G$, is affected by external factors such as the environment, the input in the system and the sensors drift. This will clearly affect the residual signal and it is handled in the following way: the set of possible environmental conditions can be reduced by using information coming from a sensor monitoring the environment and collecting response data only under a specified condition, the set of possible input signals is reduced as explained earlier (§ 4.1.1) and with the assumption that the remaining variability in the behavior of the structure is small, this can be considered as a modeling error. Furthermore, sensor drift can be managed by periodic recalibration and demeaning of the measured data so that only dynamic responses are used in the damage detection.

4.2 Algorithm to be embedded

The model based approach can be used to detect defects; now, it is necessary to develop what will be used in the lower tier of the network.

4.2.1 Lower tier architecture

At the lower tier, the local monitoring of the structure can be broken down into two steps. First, it is necessary to generate a residual signal that will estimate how far the actual behavior is from the expected one. Then, based on this residual signal, a decision must be made regarding the state of the structure.

![Figure 8 Lower tier architecture](image-url)
4.2.2 Practical residual generation

Like in the regular model-based approach in process monitoring, the residual generation relies on a model. Here we are using a model that will connect the first set of measurements $Y_1$ to the second one $Y_2$ so that in the frequency domain:

$$\tilde{Y}_2(z) = \tilde{H}_{21}(z) \cdot Y_1(z) = \sum_{k=0}^{Na} a_k \cdot z^{-k} \cdot Y_1(z) \quad \frac{1}{1 + \sum_{k=1}^{Nb} b_k \cdot z^{-k}}$$

(4.17)

The residual signal is the difference between the actual signal $Y_2$ and its estimation using $Y_1$. This idea can be illustrated by the block diagram:

$$Y_1 \quad \text{Bridge} \quad Y_2 \quad \text{Residual}$$

$\text{Model} \quad \text{Residual}$

Figure 9 Residual generation

On a practical stand point, the computation of the model’s response must be done in the time domain to reduce the number of operations. Then we can use difference equations to model the bridge’s behavior:

$$\tilde{y}_2(n) = \sum_{k=0}^{Na} a_k \cdot y_1(n-k) - \sum_{k=1}^{Nb} b_k \cdot \tilde{y}_2(n-k), \forall n > Nb$$

(4.18)

With the initialization $\tilde{y}_2(n) = y_2(n), \forall n \leq Nb$

This model leads to the following expression of the residual signal:

$$r(n) = y_2(n) - \tilde{y}_2(n)$$

(4.19)

$$r(n) = y_2(n) - \left(\sum_{k=0}^{Na} a_k \cdot y_1(n-k) - \sum_{k=1}^{Nb} b_k \cdot \tilde{y}_2(n-k)\right), \forall \max(Na, Nb) \leq n \leq N$$

(4.20)

$$r(n) = 0, \forall n < \max(Na, Nb)$$
4.2.3 Local decision making

Usually for a system monitored continuously, the signal is compared to a threshold (fixed or dynamic) to detect a change in the system behavior. In the case when the bridge is monitored intermittently with a noisy sensor, another detection algorithm should be used. As the sensor noise can be modeled as a Gaussian noise, its expectation is null. Then it is possible to reduce the effect of the noise on the residual signal and get the same information by taking the expectation of the residual signal.

\[ R(n) = \mathbb{E}[r(n)] \quad (4.21) \]

There is a problem to compute the expectation of \( r(n) \) because we do not have access to the ensemble average. In case of a non-damaged structure, the residual signal takes non-zero values only because of modeling errors and noise in the system. In addition, if the noise is assumed to be wide sense stationary, the random process \( r(n) \) can be considered wide sense stationary as well. And with the assumption that its auto covariance function \( K_r(t) \) follows

\[ \lim_{T \to +\infty} \left( \frac{1}{2T} \cdot \int_{-T}^{T} \left( 1 - \frac{|t|}{2T} \right) \cdot K_r(t) \cdot dt \right) = 0, \]

which means the function is decreasing “fast enough”, it is possible to assume the signal \( r(n) \) to be ergodic in the mean. With this assumption, the ensemble average of the residual signal for \( N \) samples can be estimated by its time average:

\[ R \approx \frac{1}{N} \sum_{n=\max(N_a,N_b)}^{N} \left[ y_2(n) - \left( \sum_{k=0}^{N_a} a_k \cdot y_1(n-k) - \sum_{k=1}^{N_b} b_k \cdot y_2(n-k) \right) \right] \quad (4.22) \]

\( R \) will be called the residual number by analogy to the residual signal.

In case of a perfect bridge’s model embedded in the device and with no noise in the measurement, a fault in the system would be detected if the residual number is different than zero. However, in the real world, some modeling error and other perturbations will occur, which can create a non-zero residual number for an undamaged system. This problem makes necessary the comparison of the residual number to a threshold. In addition, as the amplitude of the residual signal depends on the input’s
signal amplitude and as this amplitude won’t be consistent, the threshold \( h \) needs to be a function of the input signal. Here we will use the sum of the absolute value of the input signal:

\[
h = f\left(\text{Amplitude}(y_1)\right) = f\left(\frac{\sum_{n=1}^{N}|y_1(n)|}{N}\right)
\]  

(4.23)

The analytical expression of the threshold will then depend on the modeling error and the false alarm rate allowed by the user. A simple expression would be a linear threshold and the function \( f(\cdot) \) becomes a scaling coefficient:

\[
h = \alpha \cdot \sum_{n=1}^{N}|y_1(n)| = h(\alpha)
\]

(4.24)

Then the local decision making is:

\[
\text{if}|R| > h, \text{ then the system is damaged}
\]

\[
\text{otherwise, the system is safe}
\]

This provides the local report, it is a binary signal 0 safe system, 1 faulty system.

**4.2.4 Performance of the processing**

According to the equation that expresses the residual number, for \( N \) measured samples and using \( Na \) and \( Nb \) coefficients for the transfer function, this algorithm requires the following number of elementary computation.

To compute the residual number:

\[
(N - \max(Na, Nb)) \cdot (2 + Na + Nb) \text{ additions}
\]

\[
(N - \max(Na, Nb)) \cdot (Na + Nb) \text{ multiplications}
\]

To compute the threshold:

\[
N - \max(Na, Nb) \text{ additions}
\]

\[
N - \max(Na, Nb) \text{ multiplications}
\]
Then, as \( N \gg \max(Na, Nb) \), the total number of computational operation is of the order of \( N \cdot (Na + Nb) \). The order of complexity of this algorithm is \( O(N) \).

### 4.3 Simulations

Getting access to a real structure to test the performance of Structural Health Monitoring algorithms is not easy. Also, it is not easy to model a complex structure like a bridge using finite element analysis, and in order to work with some realistic signals; some huge computational capabilities would be needed. To address these constraints, the bridge’s behavior will be approximated by the one of a cantilever beam loaded and then left under free vibration. The following simulations are used as a proof of concept for the model based approach.

Figure 10 Loading scenario

#### 4.3.1 Experiment set up

*Signals are not all expressed using the International System of Units: acceleration is expressed in mini-g-force or milli-g (1 milli-g = 9.80665 × 10\(^{-3}\) m/s\(^2\)), time is expressed in seconds, force is expressed in Newton and strain is dimensionless (it is a displacement per unit length).*
Simulations are done with Matlab using a finite element model of the beam with different levels of damage. Only the two firsts modes are used here as they carry most of the information about the structure. Below are the geometric characteristics of the beam used:

![Figure 11 Geometrical characteristics of the beam modeled](image)

It is possible to simulate different levels of damage and get the acceleration and strain measurements at any point on the beam using the Finite Element model. It is important to highlight the fact that we are dealing with minor damage in the structure as it represents only 3% of the length of the beam. Even if the level of damage is 50% of the width, we have to keep in mind in the following simulation that this is still a damage at its early age.

We can see for example the difference between the measured signals in the case of no damage or 33% damage with a load of 1N for 25ms in input. The minor damage changed the shape of the modes which can be observed by looking at the peaks in the time signals below. Sensors and step function in input are at the point A.
Figure 12 Changes in the time series signals for a damaged beam
In the simulations, the input signal, a loading step, will vary in amplitude – as the weight of truck crossing the bridge is variable – and in duration – as the speed of crossing traffic will vary as well. In order to cover a wide range of operational condition, the amplitude and duration of the impulse in the input signal will be a random value uniformly distributed around the nominal conditions (load of 1N for 25ms). The load will vary from 0.5N to 1.5N and the duration of the loading will vary from 10ms to 50ms (in order not to reach a steady state).

Figure 13 Characteristics of the input signal
4.3.2 Implementation of the model based approach

In order to implement the model based approach, there are two steps to accomplish before it is possible to use the monitoring algorithm. First, it is necessary to design a model that will link the strain signal and the acceleration one for the undamaged structure. Secondly, it is necessary to find the value of the threshold \( h \) used for the decision making.

The model is determined as a “black box” using the Matlab function `ident`. The method described earlier [15] to compute the ARX coefficient could have been used here, but in order to try different models easily, the Matlab toolbox was used instead. The “black box” model uses the acceleration signal as an input \( Y_1 \) called \( u(t) \) in Matlab, and the strain signal as an output \( Y_2 \) called \( y(t) \) in Matlab. Here, after trial and error, the choice for the model used is of the form

\[
\tilde{Y}_2(z) = \sum_{k=0}^{4} a_k \cdot z^{-k} \cdot Y_1(z)
\]  

(4.25)

The coefficients are computed using the first half of the signals obtained for undamaged condition and the second half of these signal are used to validate the model. The Matlab toolbox provided us with the following results:

Discrete-time IDPOLY model: \( y(t) = B(q)u(t) + e(t) \)
\[
B(q) = -2.651e-010 (+-1.491e-020) + 3.499e-012 (+-2.583e-020) q^{-1} - 1.387e-010 (+-1.201e-020) q^{-2} + 1.808e-012 (+-2.497e-020) q^{-3} + 1.145e-010 (+-1.507e-020) q^{-4}
\]

Then, the model used to estimate the strain from the acceleration signal is:

\[
\tilde{y}_2(n) = -2.65 \cdot 10^{-10} \cdot y_1(n) + 3.5 \cdot 10^{-12} \cdot y_1(n-1) - 1.387 \cdot 10^{-10} \cdot y_1(n-2) \\
+ 1.8 \cdot 10^{-12} \cdot y_1(n-3) + 1.145 \cdot 10^{-10} \cdot y_1(n-4), \forall n > 4
\]

(4.26)

\[
\tilde{y}_2(n) = y_2(n), \forall n \leq 4
\]

And for the signal used to create to model, the error is bounded with \( \text{error}_{\max} \% \leq 1 \% \).
As a second step, the threshold needs to be determined. We are using an adaptive threshold:

\[ h = \alpha \cdot \frac{\sum_{n=1}^{N} |y_1(n)|}{N} \]  

(4.27)

The figure below illustrates, on the first row, the residual signal for many different levels of damage for inputs with different amplitudes. On the second row, the same signals are used, however, this time the residual is divided by \( h \) for \( \alpha = 1 \). This can justify the form of the threshold used.
The value for $\alpha$ should provide good fault detection capabilities as well as a low level of false positive (false alarm). We do not have enough experimental data to estimate the probability distribution of the residual number and then determine the threshold expression using a probabilistic test based on requirements regarding the False Positive (false alarm) or False Negative (missed detection) rates. Moreover, the probability distribution of the residual numbers dependent on the model used and the system monitored.

Here we are using an empirical method to determine $\alpha$. The idea is to look at the value of $R/h_{\alpha=1}$ where $R$ is the residual number and $h_{\alpha=1}$ the threshold for $\alpha = 1$ for various inputs on an undamaged beam and then choose $\alpha$ so that the probability of false alarm is set to fit the specifications. The value of $R/h_{\alpha=1}$ is computed for 100 simulations with different amplitude and durations for the loading step in and we obtain the histogram diagram shown below.
According to the experimental distribution of $R/h_{\alpha=1}$, we can suggest to use $\alpha = 2 \cdot 10^{-5}$ because it should lead to 0 false alarms with good detection capabilities.

4.3.3 Detection performances for the model-based approach

The algorithm analyzes the response for 1 second with a sampling rate of 200 Hz. This means at each simulation the algorithm is working on 200 points corresponding to about 1400 additions and 1000 multiplications total.

The system is tested under 8 levels of damage from 0% to 30%. For each level of damage, 100 analyses are made with various input signals: a step with a duration between 10ms and 50ms and amplitude 0.5 N to 1.5 N. This corresponds to the same inputs as those in the training set used to determine the threshold. Under each damage level, the percentage of fault detected is recorded and a positive value for fault detected under 0% damage means percentage of False alarm. Here, for the first tests, no noise is added to the measurements. This is not realistic even though with the use of wireless sensor networks, it is possible to install the sensor close to the phenomenon measured.
which guarantees a low signal over noise ratio, but this gives a first idea on the algorithm capabilities.

![Model Based fault detection method without noise](image)

**Figure 17 Fault detection with the model-based approach and no noise**

With this Matlab simulation, the model based approach seems to produce interesting results, especially because it is possible to detect damage at its early age.

If we add some noise into the measurements to be closer to a real situation, the algorithm might lose its detection power. The noise will be a white Gaussian noise added to the acceleration and strain signal generated by the Finite Element Analysis. The mathematical model for the noise is then a random variable with normal distribution,
mean $\mu = 0$ and variance $\sigma = 0.1$ and the signal over noise ratio is $SNR = 40dB$ which correspond to a ratio $\frac{Amplitude_{signal}}{Amplitude_{noise}} = 100$.

First the threshold is computed again, using the same method: $\alpha = 3 \cdot 10^{-4}$ for $\text{Prob(\text{False Alarm})} \leq 40\%$ which might be acceptable as the global decision is taken at a higher tier that will erase isolated “False alarms”.

![Figure 18 Determination of the scaling parameter alpha for a noisy environment](image)

Then, the system is tested under levels of damage from 0% to 70% and for each level damage, 100 analyses are made with various properties of the input signal. As expected the algorithm is less efficient but these results might be good enough for damage detection at the local level, as later on data are combined to improve the fault detection capabilities.
We can look at how the noise changed the results. For a small level of damage, the effect of the noise is more important and the false alarm rate increased because of a low “residual signal”/noise ratio. Besides, for a higher level of damage, the level of “missed detection” increased as well. Overall, the system cannot perform fault detection with a sufficient reliability to be used in the real world. In order to correct the algorithm from making these errors, some work is needed at the higher tier, where the local decision making report are combined. The key point is still the detection capability for a minor damage. As stated earlier, 50% damage on the plot correspond to 50% of the width to 3% of the length.

If the noise combined with the measurements is increased, the fault detection power of the algorithm decrease deeply as shown by the figure below where the signal over noise ratio is SNR = 20dB which correspond to a ratio $\frac{\text{Amplitude}_{\text{signal}}}{\text{Amplitude}_{\text{noise}}} = 10$. 

Figure 19 Fault detection with the model-based approach and noise with SNR=40dB
4.3.4 Comparison with the LANL method

The same set up is used with the LANL method to detect damage in the structure using acceleration signals. A database with information about nine operational conditions – because we are using nine different step durations with 1N amplitude – is constructed to compare it with the measurements from the damaged beam. The order of the AR model used is $p = 30$ and at the second step, the model used is ARX(5,5). For the simulation without noise added to the measured signals, the threshold is determined using the same type of empirical method and is set at $h = 1.0002$. For the simulation, the same input signals as for the experiments about the model-based approach are used and the results are shown below:
Figure 21 Fault detection with the LANL method and no noise

This algorithm seems to have less fault detection power as the model based approach. However, this can be explained by the limited set of operational condition stored in the database used for the simulation and because the measured signals are not totally stationary: here we are dealing with a step as input instead of a white noise as recommended by [14].

The main advantage of the method developed by the LANL is its robustness to noise. The results for the experiment with noise are very similar to those without noise with “just” an increase in the false alarm rate, as shown in Figure 22 and Figure 23.
Figure 22 Fault detection with the LANL method and noise with SNR=40dB

Figure 23 Fault detection with the LANL method and noise with SNR=20dB
Then, at the current stage, the LANL method seems to be less affected by sensing noise than the Model based approach. Even if the fault detection power of the LANL method is not fairly greater than the one of the new approach, there is an important price to pay for using the LANL method: it requires numerous data exchanges between motes and a database and greater computational capabilities needed which can be translated as an increase in the energy consumption and a reduction of the network life time.

According to this “proof of concept” experiment, the model based approach as introduced here showed some good fault detection capabilities for minor damage. Of course this capability depends on the model used and the structure monitored but it is interesting to investigate deeply the potential of this technique.

4.3.5 Complementary simulations

In order to validate the potential of the model-based approach, it is also interesting to investigate two points: (a) is it possible to detect damage with inputs outside of the training set used to determine the threshold? (b) is this method capable of handling inputs more realistic than a step?

Inputs outside of the training set. The major drawback of the current methods used to detect damage in a structure, such as the LANL method, comes from the fact that if an external condition has not been recorded in the database of “undamaged behaviors”, the resulting vibration signals will be associated with a “damaged” state even though damage may not be present. The training set used to determine alpha contains step signal with duration between 10 ms and 50ms and amplitude between 0.5N and 1.5N. The value of alpha used in the dynamic threshold is \( \alpha = 3 \cdot 10^{-4} \). For the following simulations, the duration of the input signal is set between 45ms and 75ms (refer to Figure 25) and the amplitude is set between 1.5N and 2.5N (refer to Figure 26). The signal-to-noise ratio is: \( \text{SNR} = 40 \text{ dB} \).
Figure 24 Model-based approach with input signals inside of the training set

Figure 25 Model-based approach with inputs' duration outside of the training set
Figure 26 Model-based approach with inputs’ amplitude outside of the training set

The damage detection capabilities of the model based approach have not been too much affected by the use of an input outside of the training set (compare to Figure 24); these capabilities are less dependent on the exact knowledge of the excitation of the structure. This represents a major advantage for this technique and it can be a source of motivation to extend research in this area.

**Realistic inputs.** The previous experiments are using a step function as input to approximate a vehicle crossing the bridge. A more realistic way to approximate the crossing traffic is to consider a bell-shaped signal: all the subsets of the bridge are linked together, then as a vehicle enter the bridge, the whole structure is excited and the amplitude of the force applied is closer to a bell than a step. In the following simulations, the input is a bell-shaped impulse with duration between 20ms and 100ms and amplitude between 0.5N and 1.5N (refer to Figure 27). The threshold is determined using the empirical method, $\alpha = 3.5 \cdot 10^{-4}$, and the signal is combined with a noise for SNR = 40dB.
Figure 27 Bell-shaped input signal

Figure 28 Model-based approach with bell-shaped inputs signal on the structure
The damage detection capabilities of the model-based approach for damage at earliest stage in this case are not as good as in the case where the input is a step. For a greater level of damage in the structure, the damage detection capabilities of the method are less affected by the change in the input’s shape.

These additional simulations show that it is still possible to detect damage even when the input is outside the range used to determine the threshold. Although these results show promise, more work is needed to determine the applicability of this result.

4.4 Future work on the model-based approach

4.4.1 Choosing the order of the model used

In the current approach, the order of the model used has been determined by trial and error. This method is not realistic when many smart sensors are used, especially as hypothetically hundreds of sensors could be connected in a wireless sensor network to perform structural health monitoring, and a specific method to determine the order of the model is needed. This is an open question and two approaches should be considered: (1) use of a model with high order to accommodate noise, higher mode effects, sensor drift and other perturbations, and (2) use of a low order model and leaving some degrees of freedom in the model to accommodate a wide range of operational conditions and excitations on the structure. The second point is based on the idea that the goal is to detect damage by looking at changes in the behavior of the structure.

This problem needs to be solved prior to implementing of a model-based approach for structural health monitoring. There might be a solution for some particular cases where it is possible to have some knowledge about the number of physical modes in the structure. This knowledge would give an estimation of the order to use for the model. For example, in a mass-spring-damper system, the acceleration and the force are physically linked through a second order differential equation. In this case, a second order model is appropriate.
4.4.2 Improving the robustness to noise

As observed in the previous simulation, one of the main drawbacks of the model-based approach is its sensitivity to noise. In order to solve this problem, one idea is to use more than two measurements and combine them. For example: by using three vibration signals, it is possible to generate three residual signals and average them into a new residual signal that is less affected by the noise.

It is also necessary to improve the way alpha, the scaling parameter in the threshold $h$, is determined. Here an empirical method is used and the main drawback is that, after defining the level of false alarm in the system, there is no way to estimate what will be the rate of missed detection when the system will be damaged. The idea is to model the statistical distribution of the residual number for the undamaged and damaged structure, in order to set up a hypothesis test. This will enable the user to determine alpha regarding to the levels of missed detection and false alarm he can tolerate.

4.4.3 Using all of the information available

The current decision making algorithm gives a binary result, so there is a loss of information. It should be interesting to keep the information regarding “how far are we from the undamaged behavior?” and use it directly in the data fusion process. A first step could be, for example, to determine many thresholds for the local decision making in order to add details and make a classification of the behavior between “not damaged”, “maybe damaged” and “damaged”. Another approach could be to use directly the residual number for the data fusion. As the residual number is affected by the amplitude of the input signal, this could work practically by sending a modified residual number $Rm$ with:

$$\frac{R_m}{h_{\alpha=1}} = R / h_{\alpha=1}$$  \hspace{1cm} (4.28)
Using equations (4.22) and (4.24), when \( N \) samples are considered in the analysis of the structural health, the expression of the modified residual number becomes:

\[
R_m = \sum_{n=\max(N_a,N_b)}^{N} \left( y_2(n) - \frac{\sum_{k=0}^{N_a} a_k \cdot y_1(n-k) - \sum_{k=1}^{N_b} b_k \cdot y_2(n-k)}{\sum_{n=1}^{N} |y_1(n)|} \right)
\] (4.29)

Using a modified residual number for the local report is a solution to keep as much information. This also requires a designing a way to combine these data as there is an important variability in the value that \( R_m \) can take, and the future work should address this problem.
5.1 Data fusion

At the higher tier, local reports are fused to create a global report regarding the health of the whole structure. If we look at a structure, a bridge for instance, there are very little chances that a low degree of damage on the right hand side of the structure will have an effect on the signals measured on the left hand side of the structure. Then, when data is combined, it needs to be done among data coming from similar health conditions. In other words, as the health of the structure will depend on the location, data fusion should be done locally by subsets and when a subset detects a fault, this message is spread out around the network.

The most intuitive way to define a subset following the requirements is to choose the k closest neighbors. This makes sense because these k measurements monitor subsets of the structure that will most likely present a similar level of damage. If the node density of the wireless sensor network is high enough, we can make the assumption that in each subset, all the measurements are experiencing the same level of damage.

5.2 Global decision using a voting scheme

At each mote, the local report can take two values: 0 for reporting an undamaged structure and 1 for a damaged one. The suggestion is to perform data fusion using a voting process: in each subset, each mote gets the local report from its k closest
neighbors. Each local report from itself or its neighbors count like a vote and the mote’s
global report will take the value supported by the absolute majority of motes in its
neighborhood.

\[ \text{i.e. } \sum \text{local reports} \geq \text{floor}\left(\frac{k}{2}\right) + 1 \]  

(5.1)

As soon as the mote report “damage” for its global report, this message is spread out so
that at each mote, the global report indicate “damage”. Otherwise, if no damage is
detected, the mote stays asleep, waiting for a wake up message from the lookout mote or
for an alert message from a neighbor.

![Voting scheme using the 9 closest neighbors](image)

**Figure 29 Decision and data fusion using a voting scheme**

This step aims to improve the overall fault detection power of the network by decreasing
false alarm and missed detection.

For the actual motes in the network, the data fusion and voting process can
follow the process map suggested below. The key advantage is the limited information
exchanged among the motes (horizontal axis in the process map): only one bit is sent
from one mote to its neighbors to perform the data fusion and global decision. Then
computation is done in a decentralized manner and another message can be sent either
periodically or in the case of damage.
5.3 Voting scheme and number of sensors

By intuition, when using voting on the higher tier, the global decision making could detect damage when the probability of detection at the local level is greater than 50% if the node density of the networks tends to infinity. Also this voting system should be able to “filter” individual false alarms when the false alarm rate is less than 50%.

For the previous results from the simulations using the model based approach and an additive noise in the measurements with SNR = 40dB, we would obtain the detection
characteristic presented in red dashed line below if using an infinite density of nodes in the networks:

![Model-Based fault detection method with SNR = 40dB](image)

**Figure 31 Ideal decision making at the higher tier**

However, it is not realistic to think of an “infinite density” of nodes in the networks and we can investigate the expected performance of the data fusion using k sensor node.

The problem can be modeled this way: we have k local reports; each report can take two values 0 with probability $P_0$ and 1 with probability $P_1$. $P_0$ is the conditional probability of not having detected a damage knowing the actual level of damage i.e. $P_0 = P(\text{local} = 0 | \text{damage})$ and $P_1$ is the conditional probability if having detected a damage knowing the actual level of damage, i.e. $P_1 = P(\text{local} = 1 | \text{damage})$. For example, according to the simulations, for an undamaged beam monitored in a noisy environment with SNR = 40dB, we will use $P_1 = 0.38$ and $P_0 = 1 - P_1$. 
If $D(i)$ is the random variable associated with the local report from mote $i$, the probability density function of this random variable is given by:

$$f_D(d) = P_0 \cdot \delta(d) + P_1 \cdot \delta(d - 1)$$  \hspace{1cm} (5.2)

Furthermore, we can assume these random variables $D(i); i = 1,2,\ldots,k$ to be independent and identically distributed (i.i.d.). It makes sense because what causes different local reports coming from motes monitoring the same subset are mainly modeling errors, and sensing noise which will be independent from one mote to the other. In addition, we can assume the distributions of the random variables $D(i)$ identical if we consider the damage detection probability to be only a function of the damage.

Then, as the decision making for the global report using a voting scheme is:

$$\text{if } \sum_{i=1}^{k} D(i) > \frac{k-1}{2}, \quad \text{Vote} = \text{Damage}$$

$$\text{otherwise, } \quad \text{Vote} = \text{Undamaged}$$

We need to find the probability density function of the sum of $k$ i.i.d. random variables. With the previous assumptions and if we are working in the conditional probability space, the voting problem for a $k$-subset of motes is equivalent as a Bernoulli Trail [21] where the probability of failure is $P_0$ and the number of trails is $k$. The probability of exactly $p$ success in $k$ Bernoulli trails is given by the binomial probability law:

$$P(\text{success} = p|\text{trail} = k, \text{failure} = P_0) = b(P_0,k,p) = C_p^k \cdot (1 - P_0)^{k-p} \cdot P_0^p$$  \hspace{1cm} (5.3)

i.e. $P\left(\sum_{i=1}^{k} D(i) = p\right) = C_p^k \cdot (1 - P_0)^{k-p} \cdot P_0^p$ \hspace{1cm} (5.4)

and $P\left(\sum_{i=1}^{k} D(i) > \frac{k-1}{2}\right) = \sum_{p=\left\lceil \frac{k+1}{2}\right\rceil}^{k} C_p^k \cdot (1 - P_0)^{k-p} \cdot P_0^p$  \hspace{1cm} (5.5)

Thus we can obtain the probability law for followed by the voting scheme in a $k$-subset:

$$P(\text{Vote} = \text{Damage}|P_0) = \sum_{p=\left\lceil \frac{k+1}{2}\right\rceil}^{k} C_p^k \cdot (1 - P_0)^{k-p} \cdot P_0^p$$  \hspace{1cm} (5.6)
5.4 Simulations

5.4.1 Simulations using probabilistic model

Using this previous result, it is possible to simulate how the results obtained in the simulation using the model based approach in a noisy environment with SNR = 40dB can be improved depending on how many sensors are used in the data fusion. Details on the implementation of these simulation are available in appendix C. Below are the probabilities to have an alert message in the global report as a function of the percent damage in the subset location.

![Graph showing fault detection at the hier tier using data fusion with 3, 5, 9 and 21 sensors](image)

**Figure 32 Influence of the number of sensors used in the voting scheme**

It is important to look at the interpretation of these results in the real world. In order for the data fusion to make sense, the sensors should be in the same neighborhood to be
subject to the same kind of damage. This means we would be able to aggregate local report from 5 or 6 sensors at best if we consider a reasonable number of sensors in the structure. However, here only the spatial axis is used. A way to be able to fuse data coming from a greater number of sensors would be to do it over time, and repeating measurements more frequently when some motes detected a potential damage. This is applicable because we are trying to detect the onset of a structural damage and we have some time between the first alert messages until it becomes risky to cross over the bridge.

5.4.2 Simulations using a Matlab model of the cantilever beam

After the simulations using the probabilistic model that gave us an understanding of the expected improvement due to the data fusion using the voting scheme, it is interesting to use this voting scheme with the previous finite element model of the cantilever beam. Measurements of strain and acceleration are taken at three different locations (see figure below), like if three smart sensors are installed in the structure.

![Figure 33 location of smart sensors on the beam](image)

For each location, the associated models and thresholds $h$ are determined and the signal are noisy with $\text{SNR} = 40\text{dB}$. More details about the code and the models used are available in appendix A. Using the same inputs signals as in the previous simulations
(§4.3), 100 experiments are performed and at each time the local reports from the three smarts sensors are combined using the voting scheme. The results are shown in Figure 34: the data fusion helped to decrease the false alarm rate, but there is not a great improvement regarding the damage detection. Part of it can be explained by the fact that when there is a “high” level of damage (relative to damage at its earlier stage) if a smart sensor misses the detection it is most likely due to the external condition, and it might affect the other sensors in the neighborhood.

![Figure 34 Data fusion using voting scheme for three smart sensors](image)

In order to include more local reports when performing the data fusion, it is possible to combine these report overtime. In the following simulation (Figure 8), results from the previous simulations are combined over 1, 2, 3, 4 and 5 experiments. Thus 3, 6, 9, 12 and 15 local reports have been combined using the voting scheme for each experiment.
Figure 35 Data fusion using the voting scheme for three smart sensors overtime

Here, there is a real improvement of the damage detection capabilities as well as a clear decrease of the fault alarm rate. These results are close to the one obtained by simulating the voting scheme previously (§5.3). The reason is that, by combining local reports over time, the actual process is close to the one modeled. Especially, the assumption regarding the independence of the random variables $D(k)$ makes more sense when the data are combined over time as in this case the local report are the outcomes of independent experiences (different excitation signals in the structure). It is interesting to point out the important decrease of the false alarm rate even when combining data over only two experiments (six local reports combined). This simulation proves that the concept of a voting scheme works well for this application.
5.5 Future work on the global decision making

The second layer in the network architecture can improve the detection power of the local model-based approach for structural health monitoring. However, there are some opportunities to improve this basic scheme as we don’t use all the information available when data is fused. On the first hand, only two states of the structure are considered: “damaged” or “undamaged”. We can expect better results if instead we consider many choices “undamaged”, “maybe damaged”, “most likely damaged” and “damaged”, for example. Or even better, if we could have a weighted metric that would represent how bad the damage seems and how confident we can be about the local report. On the other hand, we could also consider the location issuing alert messages and use some knowledge about the system’s behavior in order to dismiss some false alarms. For example, we can have the following constraint: if an alert message comes from a spot in the structure, then we should also get an alert message from the spot next to it, otherwise this is a false alarm. This can be done using some results of the “Causal Analysis” field which is part of the process monitoring area [3].
6.1 From simulation to implementation

This work introduces a new approach to fault detection using wireless sensor networks with an emphasis on the application for bridges in a remote area. For being able to set up tests with a real structure, some specific constraints need to be addressed prior to the implementation phase.

First, what should be the location of sensors? At the lower tier, the measurements from two heterogeneous sensors are combined to perform local fault detection. The dynamic relation between these two measurements is used and then in order to give a physical meaning to this relation, the sensors need to be positioned so that the energy in the system will go from the location of the measurement used as input, to the one of the measurements used as output in the model-based approach. For example this idea about the location of the sensors is illustrated in the figure below.

Second, it has been shown that a high density of sensor nodes in the network increase the fault detection power of a single node. Then the goal will be to increase the density as much as possible according to the cost constraints. One great advantage of using a wireless sensor network comes from the scalability of such network; there is then
a great advantage to use this property extensively maybe up to the point of making the following vision a reality:

“Millions of tiny sensors embedded in concrete to detect damages in bridges”

![Figure 36 Location of the sensors](image)

For the actual implementation, it is not realistic to compute the model and to determine the value of the threshold offline for each smart sensor. It is possible to automate the set up of the model-based approach on the network; this would work in two steps: (1) computation of the model by each smart sensor using model identification techniques, and (2) determination of alpha for threshold $h$. For the computation of the model, an ARX model can be used and the equation (2.4) can be solved online. To find the value of the scaling parameter alpha, it is necessary to record the value of $R/h_{\alpha=1}$ for vibration signals measured on the undamaged structure under many different operational conditions. Thus, the set up of the network requires enough memory in each smart sensor to store at least 100 values of $R/h_{\alpha=1}$ in addition to the model parameters and ID numbers of the mote in its neighborhood. After the set up of the network, this memory space needs to be allocated to store the local report and those from smart sensors in its neighborhood in order to perform the data fusion using local reports from the closest neighbors over many analyses.
6.2 Future work

During the previous chapters, many suggestions have been made to improve the model-based approach to damage detection as exposed and here are the next steps to consider:

First, there are many ways to improve the local damage detection. By increasing the number of measurements combined, it should be possible to have a method less sensitive to modeling error and noise. For example with three sensors, it is possible to have three “input/output” relations and then generate three residual signals that can be combined. Also, there is a need to model the statistical distribution of the residual number to design a probabilistic test and set the value of the threshold $h$ by using an analytical expression for a level of false positive and missed detection fixed. Furthermore, here the dynamic threshold was depending on the amplitude of the “input” signal. Another approach could be to adapt the threshold depending on the local reports from the previous analysis from other motes in the network. For example if a damage is detected by a mote, for its next analysis of the structure’s health, the same mote can take a less sensitive threshold to validate the previous local report and make sure it is not a false detection positive.

Second, there are some ideas that should improve the data fusion at the higher tier of the architecture. Here we considered data fusion among sensor nodes localized in a same neighborhood. Because of the limitation in the density of the network, it would be hard to aggregate data coming from more than 5 or 6 sensor nodes and still considering these nodes subject to the same level of damage. One solution could be then to fuse data by using the voting scheme also over time, as it was suggested earlier. In addition, here we are combining binary signal, which means that we don’t use all the information available. In the future work, it could be interesting to develop a data fusion scheme that would combine more complex local reports which express how far the system is from its actual behavior.
6.3 Conclusions

In this research, a new approach for structural health monitoring using wireless sensor networks has been developed. The idea is to use some knowledge about the structure’s behavior to verify if the actual signals measured are associated with healthy structural condition. This technique might be a way to meet the requirements of a low cost and reliable damage detection system that could be used to support bridge inspectors in remote areas. As the damage detection performance of an algorithm may vary depending on the structure monitored or the model used, more experiments and especially a field test would be needed to validate this new method.

The present work contributes the problem formulation using heterogeneous sensors and a two-tiered network architecture, the detection scheme at each node using weighted thresholds and the data fusion at the higher level, and the extensive simulations based on a finite element model of a cantilever beam. The goal has been to combine ideas from many fields in order to set the foundations for an approach in damage detection using wireless sensors networks.

There are still many challenges that need to be addressed before considering a model based approach for damage detection in a practical application, but it is hoped that the work presented here will be a first step toward a new kind of structural health monitoring system.
A.1 Local damage detection using model-based approach

% Local structure monitoring
clear
% System constants
T0=10;
N=2000;
dt=T0/(N-1);
t=dt*[1:N];

% Initialization
DD=[0 0.1 0.2 0.3 0.5 0.7] %damage considered on the beam
damgepercent=[];
Threshold1=3*10^(-4);
Threshold2=2.5*10^(-4);
Threshold3=3*10^(-4);
sim=100; % number of simulation for each level of damage

randomN=rand(1,sim);
randomNRJ=rand(1,sim);
vote=[zeros(length(DD),sim)];
vote1=[zeros(length(DD),sim)];
vote2=[zeros(length(DD),sim)];
vote3=[zeros(length(DD),sim)];

for jj=1:length(DD) % for each damage level
    dam=DD(jj) % actual damage considered in the loop
damged(jj)=0; % number of detection of damaged structure in this loop
damged1(jj)=0; % number of detection of damaged structure by mote 1
damged2(jj)=0; % number of detection of damaged structure by mote 2
damged3(jj)=0; % number of detection of damaged structure by mote 3
    for sss=1:sim % for each simulation
        n=5+round(7*(randomN(sss)-0.5)); % random duration of the step
        nrj=(0.5+randomNRJ(sss)); % random amplitude of the step

        % Simulation code

        if dam>=n % if the simulation satisfies the condition
            vote(jj,sss)=1; % increment vote for the detected damage level
            damged(jj)=damged(jj)+1; % increment damage detection count
            damged1(jj)=damged1(jj)+1; % increment damage detection by mote 1
            damged2(jj)=damged2(jj)+1; % increment damage detection by mote 2
            damged3(jj)=damged3(jj)+1; % increment damage detection by mote 3
        end
    end
end
INP=[ones(1,n) zeros(1,N-n)]*nrj;

% creation of signals
[strain1, strain2, strain3, acc1, acc2, acc3]=multidamagebeam(dam,INP);

% Sensing time
Tlim=1;
Nlim=ceil(Tlim/dt);

% Sensing noise Noise -40dB (99% of the noise is between 0 and 1/100 of the signal amplitude with Gaussian distribution)
acc1=acc1+0.01*(acc1).*randn(size(acc1))*sqrt(0.1);
strain1=strain1+0.01*(strain1).*randn(size(strain1))*sqrt(0.1);
acc2=acc2+0.01*(acc2).*randn(size(acc2))*sqrt(0.1);
strain2=strain2+0.01*(strain2).*randn(size(strain2))*sqrt(0.1);
acc3=acc3+0.01*(acc3).*randn(size(acc3))*sqrt(0.1);
strain3=strain3+0.01*(strain3).*randn(size(strain3))*sqrt(0.1);

% Residual Generation
% Mote 1

% % Model
% % Discrete-time IDPOLY model: y(t) = B(q)u(t) + e(t)
% % B(q) = -2.651e-010 (+-1.491e-020) + 3.499e-012 (+-2.583e-020) q^-1 - 1.387e-010 (+-1.201e-020) q^-2 +
% % 1.808e-012 (+-2.497e-020) q^-3 + 1.145e-010 (+-2.497e-020) q^-4
% %
%
d=10+n; %delay before residual generation

inp=acc1(1:Nlim);
outp=[];
for i=1:d
    outp(i)=strain1(i);
end
for i = d+1:Nlim
    outp(i)=-2.651e-010*inp(i)+ 3.499e-012*inp(i-1)- 1.387e-010*inp(i-2) + 1.808e-012*inp(i-3)+ 1.145e-010*inp(i-4);
end
residual1 = (strain1(1:Nlim)-outp);
finalresidual1=sum(residual1)/length(residual1);
ampInput1 =sum(abs(strain1(1:Nlim)))/length(strain1(1:Nlim));

% Decision making
if abs(finalresidual1)>Threshold1*ampInput1 % damage detected
damaged1(jj)=damged1(jj)+1;
vote1(jj,sss)=1;
end

% Mote 2
% Model
% Discrete-time IDPOLY model: y(t) = B(q)u(t) + e(t)
% B(q) = -2.735e-008 (+-1.014e-020) + 3.781e-009
% (+-1.754e-020) q^-1 - 1.431e-008 (+-7.935e-021) q^-2
% - 1.347e-009 (+-1.655e-020) q^-3 + 1.178e-008
% (+-9.914e-021) q^-4

\[
d = 10 + n; \text{ delay before residual generation}
\]

\[
\text{inp} = \text{acc2}(1: \text{Nlim});
\]
\[
\text{outp} = []; 
\]
\[
\text{for i} = 1:d
\]
\[
\text{outp(i)} = \text{strain2}(i);
\]
\[
\text{end}
\]
\[
\text{for i} = d+1: \text{Nlim}
\]
\[
\text{outp(i)} = -2.735e-008*\text{inp(i)} + 3.781e-009*\text{inp(i-1)} - 1.431e-008*\text{inp(i-2)} - 1.347e-009*\text{inp(i-3)} + 1.178e-008*\text{inp(i-4)};
\]
\[
\text{end}
\]
\[
\text{residual2} = (\text{strain2}(1: \text{Nlim}) - \text{outp});
\]
\[
\text{finalresidual2} = \text{sum} \left( \text{residual2} \right)/\text{length} \left( \text{residual2} \right);
\]
\[
\text{ampInput2} = \text{sum} \left( \text{abs} \left( \text{strain2}(1: \text{Nlim}) \right) \right)/\text{length} \left( \text{strain2}(1: \text{Nlim}) \right);
\]
\[
\text{R3} = [\text{R3 abs} \left( \text{finalresidual2}/\text{ampInput2} \right)];
\]
\[
\text{if abs} \left( \text{finalresidual2} \right) > \text{Threshold2} \times \text{ampInput2} \quad \text{damage detected}
\]
\[
\text{damaged2(jj)} = \text{damged2(jj)} + 1;
\]
\[
\text{vote2(jj, sss)} = 1;
\]
\[
\text{end}
\]

% Mote 3
% Model
% Discrete-time IDPOLY model: y(t) = B(q)u(t) + e(t)
% B(q) = -1.198e-007 (+-1.948e-020) + 4e-008 (+-3.383e-020)
% q^-1 - 6.271e-008 (+-1.567e-020) q^-2 - 1.641e-008
% (+-3.331e-020) q^-3 + 5.14e-008 (+-2.009e-020) q^-4
% %

\[
d = 10 + n; \text{ delay before residual generation}
\]

\[
\text{inp} = \text{acc3}(1: \text{Nlim});
\]
\[
\text{outp} = []; 
\]
\[
\text{for i} = 1:d
\]
\[
\text{outp(i)} = \text{strain3}(i);
\]
\[
\text{end}
\]
\[
\text{for i} = d+1: \text{Nlim}
\]
\[
\text{outp(i)} = -1.198e-007*\text{inp(i)} + 4e-008*\text{inp(i-1)} - 6.271e-008*\text{inp(i-2)} - 1.641e-008*\text{inp(i-3)} + 5.14e-008*\text{inp(i-4)};
\]
\[
\text{end}
\]
\[
\text{residual3} = (\text{strain3}(1: \text{Nlim}) - \text{outp});
\]
\[
\text{finalresidual3} = \text{sum} \left( \text{residual3} \right)/\text{length} \left( \text{residual3} \right);
\]
\[
\text{ampInput3} = \text{sum} \left( \text{abs} \left( \text{strain3}(1: \text{Nlim}) \right) \right)/\text{length} \left( \text{strain3}(1: \text{Nlim}) \right);
\]
\[
\text{R3} = [\text{R3 abs} \left( \text{finalresidual3}/\text{ampInput3} \right)];
\]
\[
\text{if abs} \left( \text{finalresidual3} \right) > \text{Threshold3} \times \text{ampInput3} \quad \text{damage detected}
\]
damaged3(jj)=damaged3(jj)+1;
vote3(jj,sss)=1;
end

% Voting scheme
if (vote1(jj,sss)+vote2(jj,sss)+vote3(jj,sss))>=2
    damaged(jj)=damaged(jj)+1;
vote(jj,sss)=1;
end
end

damagepercent(jj)=100*damaged(jj)/sim;
damagepercent1(jj) = 100*damaged1(jj)/sim;
damagepercent2(jj) = 100*damaged2(jj)/sim;
damagepercent3(jj) = 100*damaged3(jj)/sim;
end

a=3
csvwrite(['vote1_' num2str(a) '.dat'],vote1 ) % Save data for global decision making in § A.4
csvwrite(['vote2_' num2str(a) '.dat'],vote2 )
csvwrite(['vote3_' num2str(a) '.dat'],vote3 )
csvwrite(['vote_' num2str(a) '.dat'],vote )

figure(1)
plot(100*DD,damagepercent,'.-')
title(['Percentage of fault detection for various level of damage' ])
xlabel('% damage of the beam')
ylabel('% fault detection')
axis([0 max(100*DD) 0 100])

figure(2)
plot(100*DD,damagepercent1,'.-')
hold on
plot(100*DD,damagepercent2,'.-', 'color','r')
plot(100*DD,damagepercent3,'.-', 'color','k')
hold off
title(['Percentage of fault detection for various level of damage' ])
xlabel('% damage of the beam')
ylabel('% fault detection')
axis([0 max(100*DD) 0 100])

A.2 Function multidamagebeam

function [strain1, strain2, strain3, acc1, acc2, acc3]=multidamagebeam(dam,INP)
damaged=1-dam;
k=[];
m=[];

N=101;% number of nodes
NE=100;% number of elements
Area=121*10^-6; % Area of the beam section
EI=16.09;% EI of the beam section
rho=2700;% density of the beam per unit
nn=[1:N;
    2:N+1]; % name the dofs of each nodes
L=0.49;% length of the beam
h=L/NE;% length of elements

begin=45;
ends=55;
% lengthdamaged=h*(ends-begin)*100 % damaged length in centimetre
EID=16.09*damaged;
AreaD=(121*10^-6);
codes=zeros(2,N);
codes(:,1)=[1;1];
codes=reshape(codes,2*N,1); % find the dofs which is rigid

k=zeros(2*N,2*N);
m=zeros(2*N,2*N);
for e=1:NE
    nne=nn(:,e);
    map_e=[2*nne(1)-1;2*nne(1);2*nne(2)-1;2*nne(2)];
    if e>begin&e<ends
        ke=(EID/h^3).*[12 6*h -12 6*h;6*h 4*h^2 -6*h 2*h^2;
                      -12 -6*h 12 -6*h;6*h 2*h^2 -6*h 4*h^2];
        me=(rho*AreaD*h/420)*[156 22*h 54 -13*h;22*h 4*h^2 13*h -3*h^2;
                               54 13*h 156 -22*h;-13*h -3*h^2 -22*h 4*h^2];
    else
        ke=(EI/h^3).*[12 6*h -12 6*h;6*h 4*h^2 -6*h 2*h^2;
                      -12 -6*h 12 -6*h;6*h 2*h^2 -6*h 4*h^2];
        me=(rho*Area*h/420)*[156 22*h 54 -13*h;22*h 4*h^2 13*h -3*h^2;
                               54 13*h 156 -22*h;-13*h -3*h^2 -22*h 4*h^2];
    end
    k(map_e,map_e)=k(map_e,map_e)+ke;
    m(map_e,map_e)=m(map_e,map_e)+me;
end
% this loop is to generate globel matrix from local matrix
en=find(codes==0);
rn=find(codes==1);
kr=k(en,en);
mr=m(en,en);
[PHI,Omegaz]=eig(kr,mr);
% eliminate the rigid dofs from k and m matrix to simplify the
% calculation
U0=-0.01*zeros(2*NE,1);
uu0=inv(PHI)*U0;
K=2; % use 4 modes to simulate the vibration
M=PHI(:,[1:K])'*mr*PHI(:,[1:K]);
C=zeros(K,K);
gg=0.014;
for i=1:K
    C(i,i)=2*M(i,i)*sqrt(Omegaz(i,i))*gg;
end
for n=1:K
    wn=sqrt(Omegaz(n,n));
    gn=PHI(:,n)'*ones(2*(N-1),1);
    [q,a]=newmark(wn,gn,M,C,n,uu0,INP);
    Q(n,:)=q;
    QA(n,:)=a;
end
u=PHI(:,[1:K])*Q;
acc=PHI(:,[1:K])*QA;

T0=10.00;
N=2000;
dt=T0/(N-1);
t=dt*[1:N];

strain1=(u(200,:)-u(198,:))*4.8/4.9;
acc1=acc(199,:)';

strain2=(u(190,:)-u(188,:))*4.8/4.9;
acc2=acc(189,:)';

strain3=(u(180,:)-u(178,:))*4.8/4.9;
acc3=acc(179,:)';

A.3 Function newmark

function [q,a]=newmark(wn,gn,M,C,n,uu0,INP);
m=M(n,n); % refine it
k=m*wn^2;
c=C(n,n);
T0=10.00;
N=2000;
dt=T0/(N-1);
t=dt*[1:N];

randn('state',sum(100*clock))
P=gn*INP'; %gn*randn(N,1); % if P=1, the force 1 N
p=P';
u(1)=uu0(n,1);
v(1)=0;
%p(1)=gn*1000*randn(1,1);
a(1)=(p(1)-k*u(1))/m;
g=0.5;
b=1/6;

for i=2:N
    kh=m/b/dt^2+g*c/b/dt+k;
b1=m/b/dt+g*c/b;
b2=m/2/b+dt*c*(g/2/b-1);

dph=p(i)-p(i-1)+b1*v(i-1)+b2*a(i-1);
 du=dph/kh;
 dv=g*du/b/dt-g*v(i-1)/b+(1-g/2/b)*dt*a(i-1);
 da=du/b/dt^2-v(i-1)/b/dt-a(i-1)/2/b;

 u(i)=u(i-1)+du;
 v(i)=v(i-1)+dv;
 a(i)=a(i-1)+da;

end

q=u;

A.4 Data fusion using voting scheme

% Global damage detection
vote1=csvread('vote1_1.dat');%load data from §A.1: local damage detection with
model-based approach
vote2=csvread('vote2_1.dat');
vote3=csvread('vote3_1.dat');
vote=csvread('vote_1.dat');

for n=1:5
    for dam=1:min(size(vote))
        for i=1:(length(vote1)-n)
            if (sum([vote1(dam,i:i+n-1) vote2(dam,i:i+n-1) vote3(dam,i:i+n-1)]))>3*n/2)
                globalvote(n,dam,i)=1;
            else globalvote(n,dam,i)=0;
            end
        end
    end
damagedetection(n,dam)=100*sum(globalvote(n,dam,:))/length(globalvote(n,dam,:));
end

damagepercent=[0 0.1 0.2 0.3 0.5 0.7];

figure(1)
plot(100*damagepercent,damagedetection)
xlabel('% damage of the beam')
ylabel('% fault detection')
axis([0 70 0 100])
grid on
title(['Fault detection at the hier tier using data fusion with ' num2str(1:5) ' experiments'])

figure(2)
for j=1:min(size(vote))
damagepercent1(j)=sum(vote1(j,:))
damagepercent2(j)=sum(vote2(j,:))
damagepercent3(j)=sum(vote3(j,:))
end
plot(100*damagepercent,damagepercent1,'.-')
hold on
plot(100*damagepercent,damagepercent2,'.-','color','r')
plot(100*damagepercent,damagepercent3,'.-', 'color','k')
plot(100*damagepercent,damagedetection(1,:), '--','color','r')
hold off
title(['Percentage of fault detection for various level of damage'])
xlabel('% damage of the beam')
ylabel('% fault detection')
grid on
axis([0 max(100*damagepercent) 0 100])
B.1 Damage detection using the LANL method

% Damage detection using the LANL method
% System constants
T0=10;
N=2000;
dt=T0/(N-1);
t=dt*[1:N];

dam=0;

Tlim=0.5;
Nlim=ceil(Tlim/dt);
d=30;

% % % % % % % % % % %
% Set up %
% % % % % % % % % % %

nn=[2:9]; % different duration for each operational condition
database=[zeros(1,46)];
for i=1:length(nn)
    n=nn(i);
    T0=10;
    N=2000;
    dt=T0/(N-1);
    t=dt*[1:N];
    INP=[ones(1,n) zeros(1,N-n)];
    [strain,acc]=damagebeam(dam,INP);
    % data normalisation
    x=(acc(d+1:Nlim)-mean(acc(d+1:Nlim)))/sqrt(var(acc(d+1:Nlim)));
    t=t(d+1:Nlim);
    x=x';
    [phi, alpha, beta, sig_xepsilon]=setup(x,Nlim,d,t);
```matlab
database(i,:)=[i n length(phi) phi' length(alpha) alpha' length(beta) beta'
sig_xepsilon];
end
csvwrite('Xdatabase',database) % saving of the database

database=csvread('Xdatabase'); % loading of the database
armaX=database(:,1:3+database(1,3));

%%%%%%%%%%%
% Damage detection %
%%%%%%%%%%%
T0=10;
N=2000;
dt=T0/(N-1);
t=dt*[1:N];

DD=[0 0.1 0.2 0.3 0.5 0.7]; % damage considered on the beam
sim=100; % number of simulation for each level of damage

Threshold=1.01; % 1.2692;

randomN=rand(1,sim);
randomNRJ=rand(1,sim);
for jj=1:length(DD)
dam=DD(jj);
damaged(jj)=0;
for sss=1:sim
T0=10;
N=2000;
dt=T0/(N-1);
t=dt*[1:N];
n=5+round(7*(randomN(sss)-0.5));
nrj=1*(0.5+randomNRJ(sss));
INP=[ones(1,n) zeros(1,N-n)]*nrj;
[strain,acc]=damagebeam(dam,INP);

% Sensing noise Noise -20dB variance 0.1 mean 0
acc=acc+0.01*(acc).*randn(size(acc))*sqrt(0.1);

% Sensing time
Tlim=0.5;
Nlim=ceil(Tlim/dt);

% process monitoring %
% data normalisation
y=(acc(d+1:Nlim)-mean(acc(d+1:Nlim)))/sqrt(var(acc(d+1:Nlim)));
t=t(d+1:Nlim);
y=y';
% compute AR coefficients
N_=Nlim-d;
p=30;
[phi_y]=arma(y,p);
```

y\_est = y(1:p);
for i = p+1:N_
y\_est(i) = fliplr(y\_est(i-p:i-1))*phi\_y;
end

ey = y - y\_est;

% find operational condition
for j = 1:min(size(armaX))
    Dif(j) = ((armaX(j,4:length(armaX)))' - phi\_y)'*((armaX(j,4:length(armaX)))' - phi\_y);
end
[t condition] = min(Dif);

% load data for this operational condition from the database
n\_alpha = database(1,3)+5;
a = database(1, database(1,3)+4);
n\_beta = n\_alpha + a + 1;
b = database(1, n\_alpha + a);
alpha\_cond = [database(condition,n\_alpha:n\_alpha+a-1)]';
beta\_cond = [database(condition,n\_beta:n\_beta+b-1)]';

y\_est2 = y(1:max(a,b));
for i = max(a,b)+1:N_
y\_est2(i) = fliplr(y\_est(i-a:i-1))*alpha\_cond + fliplr(ey(i-b:i-1))*beta\_cond;
end

epsilon\_y = y - y\_est2;
sig\_epsilon\_y = var(epsilon\_y);
sig\_xepsilon = database(condition,46);

% Decision making
nu = sig\_epsilon\_y / sig\_xepsilon;
if nu > Threshold % damage detected
damaged(jj) = damaged(jj) + 1;
end

damagepercent(jj) = 100*damaged(jj)/sim;
end
figure(6)
plot(100*DD, damagepercent, '.-')
title(['Percentage of fault detection for Threshold = ' num2str(Threshold) ' for various level of damage '])
xlabel('% damage of the beam')
ylabel('% fault detection')
axis([0 max(100*DD) 0 100])

\textbf{B.2 Function Setup}

function [phi\_x, alpha, beta, sig\_xepsilon] = setup2(x,Nlim,d,t)
% compute AR coefficients
N = Nlim - d;
p = 30;
[phi\_x] = arma(x,p);
x\_est = x(1:p);

for i= p+1:N
    x_est(i)=fliplr(x_est(i-p:i-1))*phi_x;
end
% Compute modeling error
e=x-x_est;
% compute ARX coefficients
a=5;
b=5;
[alpha, beta]=arexo(x,e,a,b);
x_est2=x(1:max(a,b));
for i=max(a,b)+1:N
    x_est2(i)=fliplr(x_est(i-a:i-1))*alpha +fliplr(e(i-b:i-1))*beta;
end
% Compute second modeling error
epsilon = x-x_est2;
sig_xepsilon=var(epsilon);

B.3 Function arexo

function [alpha, beta]=arexo(y,u,a,b)
N=length(y);
n=max(a,b)+1;
A=zeros(a+b,a+b);
for i=n:N
    A=A+[fliplr(y(i-a:i-1)) fliplr(u(i-b:i-1))]'*[fliplr(y(i-a:i-1)) fliplr(u(i-b:i-1))];
end
B=zeros(a+b,1);
for i=n:N
    B=B+[fliplr(y(i-a:i-1)) fliplr(u(i-b:i-1))]'*y(i);
end
SOL=(A^(-1)*B);
alpha=SOL(1:a);
beta=SOL(a+1:a+b);

B.4 Function arma

function [phi]=arma(x,p)
rr=xcorr(x,x,'unbiased');
R=rr(((length(rr)+1)/2:1:length(rr)));
for i=1:p
    for j=1:p
        A(i,j)=R(abs(i-j)+1);
    end
end
for j=1:p
    b(j)=R(j+1);
end
b=b';
phi=(A^(-1))*b;
% Higher tier voting system
% data from the simulations
faultdetection=[38 38 50 58 77 88 96 98 100];
damagepercent=[0 10 15 20 25 30 40 50 60 70];
K=[3 5 9 21]; % size of the neighborhood
for j=1:length(K)
    k=K(j);
    for i=1:length(faultdetection)
        P=faultdetection(i)/100;
        faultdetectionK(j,i)=100*election(P,k);
    end
end

figure(1)
plot(damagepercent,faultdetection)
xlabel('% damage of the beam')
ylabel('% fault detection')

axis([0 70 0 100])
grid on
hold on
plot(damagepercent,faultdetectionK,'r-')
hold off
title(['Fault detection at the hier tier using data fusion with ' num2str(K) ' sensors'])

function [vote]=election(P,k)
% k = size of the cluster
% P = Po, probability of no damage detected
for i=(k+1)/2:k
    prob(i)=nchoosek(k,i)*((1-P)^(k-i))*(P^i);
end
vote=sum(prob);
LIST OF REFERENCES


