Low-cost, Efficient Output-only Infrastructure Damage Detection with Wireless Sensor Networks

William Contreras & Sotirios Ziavras

***Abstract*—Structural health monitoring (SHM), which is the process of gauging the health of instrumented structures, is becoming increasingly important as much of the world's transportation infrastructure ages and deteriorates. Wireless sensor networks (WSNs) have the potential to deliver continuous SHM at a low cost. We present a novel WSN system that monitors ambient structural vibrations and looks for deviations from a baseline response to assess the condition of the structure under observation. Efficiency is achieved by implementing our damage assessment process with an innovative wireless-sensor based technique involving look-up tables. By using look-up tables we are able to minimize computations. Only 4-6 integer and two floating-point operations are required per damage assessment. This is in contrast to the majority of other methods that rely on precise engineering models of the monitored structures. Our look-up table technique is benchmarked against the brute force approach, which involves solving equations, by measuring the execution time of each implementation on a low-power Microchip microcontroller. Here we were able to achieve a speedup of roughly five. In addition, we investigate a tradeoff between average transmission power and the number of measurements required to make a decision as to the structure's condition. Finally, we benchmark our damage assessment process using a case study to verify its efficacy for bridges.**

***Index Terms*—Distributed algorithm, statistical analysis, vibration measurement, wireless sensor network.**

I. Introduction

STRUCTURAL health monitoring (SHM), which is the process of gauging the health of instrumented engineering structures, has received a great deal of attention in research in recent years. This technology is becoming increasingly important as much of the world's transportation infrastructure ages and deteriorates. Additionally, funds to repair and replace infrastructure tend to be scarce. Given this, it would be greatly beneficial to extract the maximum life out of existing infrastructure. SHM systems can be used very effectively toward this end. Because cost is a concern, however, SHM systems must be inexpensive.

Wireless sensor networks (WSNs) are the natural choice for an SHM system, as installation and maintenance costs can be quite low. A substantial amount of research has gone into using them for this purpose. Using WSNs, a number of different parameters such as strain and acoustic emission can be monitored to detect structural damage. The parameter that is most commonly monitored is vibration [1-6]. This is the parameter that we monitor as well. To eliminate the need to use controlled stimuli (e.g. a controlled live load in the case of a bridge), we monitor ambient structural vibrations. A difficulty with using ambient vibrations is that they are subject to environmental and operational variability [7]. To circumvent this problem, we treat statistical features calculated from raw vibration data from sensors placed throughout the structure of interest as random variables. The term statistical feature is used to refer to a quantity calculated from raw data that provides useful information about the data as a whole, such as a time domain average. During a training phase, we determine the joint probability distributions for pairs of these statistical feature random variables for a healthy structural condition. During an operational phase, we measure ambient vibrations and monitor the relative frequency of outliers from the healthy distributions to assess the condition of the structure. Overall then, our system is output-only: it uses only the measured vibrations to make decisions about the condition of the structure, and does not require detailed engineering knowledge of the structure. Given this, the system can be easily deployed on a wide range of structures, with little to no modification.

II. Previous Work

Approaches to the problem of SHM can be divided into two categories: input-output and output-only. With input-output approaches, values of environmental and operational parameters are determined and then used to remove the parameters' effect from the raw data. In [8], the authors pre-process structural strain data to remove as much of the effect of environmental and operational factors as possible, and use Stewhart charts to determine whether or not damage has occurred. In [9], the authors develop a prognostics methodology that consists of a joint state-parameter estimation problem. The difficulty with input-output methods is that it can be very difficult to determine the values of environmental and operational factors.

With output-only approaches, statistical methods are used to determine structural condition. As such, the values of environmental and operational parameters are not required. One

output-only technique that has been investigated is factor analysis [10, 11]. Another popular approach is to use principal component analysis [12-16]. In [17], to compensate for environmental and operational variability, an output-only method involving regression analysis is proposed. The authors of [18] suggest using a spatio-temporal infinite impulse response filter to filter out the environmental and operational effects from damage estimates. In [19], the authors employ hidden Markov models utilizing a Gaussian probability distribution to perform anomaly detection on gas turbines. The authors of [6] use transmissibility as a damage sensitive feature. In [20], the authors use Welch's t-test to look for changes in the distribution of statistical features over time to determine if structural damage has occurred. The usage of autoregressive models has also been investigated extensively [21-26]. The authors of [27] take a similar approach to that discussed in [20]. Overall, these approaches are promising with regard to achieving environmental and operational variability but they tend to be more computationally intensive than our approach, as we discuss in Section IV-C. In [28], an extensive review of vibration-based condition monitoring is presented.

To the best of our knowledge, the least computationally intensive approach that has been proposed thus far can be found in [29]. In [29], the authors propose a WSN SHM system wherein a pattern matching technique is applied to vibration data from sensor nodes around the WSN to efficiently detect damage. Although it is very efficient, the system does not incorporate a means whereby to determine the location of damage. The system that we are currently proposing builds upon that of [29] by providing a mechanism to determine which sensor node the potential damage is closest to. This is described in detail in Section IV-A. In addition to this, as discussed in Section IV-B, our current method builds upon [29] by providing a means to interpolate healthy patterns (which might not have been recorded during the training phase) in order to minimize the likelihood of false positives. Through the use of an efficient technique involving look-up tables, it does this at only a slight increase in computational cost: two floating point divisions are added. This is in contrast to the brute-force approach involving calculations, which we show (on a Microchip microcontroller) would take five times as long to execute than the look-up table approach. In the paper, we also discuss a tradeoff between average transmission power and the number of measurements required to make a decision as to the structure's condition, and benchmark our damage assessment process using a case study to verify its efficacy for bridges.

The paper is structured as follows: in Section III we discuss the topology of the WSN, in Section IV we present the details of our SHM algorithm, in Section V we analyze the computational complexity of our algorithm, in Section VI we discuss a tradeoff between average transmission power and measurement window size, and in Section VII we present our experimental results.

III. WSN Topology

Overall, our system will consist of numerous inexpensive motes distributed about the structure of interest. The motes will be divided into clusters over which structural vibrations are correlated. Such spatial correlation is common in WSN applications, and in SHM applications in particular [30, 31]. There will be a cluster head for each cluster and a single base station for the network. Each cluster will run a separate instance of the operational phase of the algorithm. The cluster head of each cluster will be responsible for performing most of the processing required during the operational phase. This eliminates the need to send a large amount of data over the network to the base station. The cluster heads will also be responsible for performing inter-cluster communication. For instance, if a cluster head detects potential damage, it will send an alert by way of inter-cluster communication to the base station. The base station, in turn, will forward the alert to the engineering office in charge of the structure. The engineering office will use the alerts to manage the deterioration of the structure such that the impact of the deterioration on the structure's users is minimal. In the case of a bridge, for example, the office might either fix the damage outright or, if funds are not available to do this, impose load limits to ensure that the bridge is safely trafficked. An illustration of the topology of the network can be seen in Fig. 1.

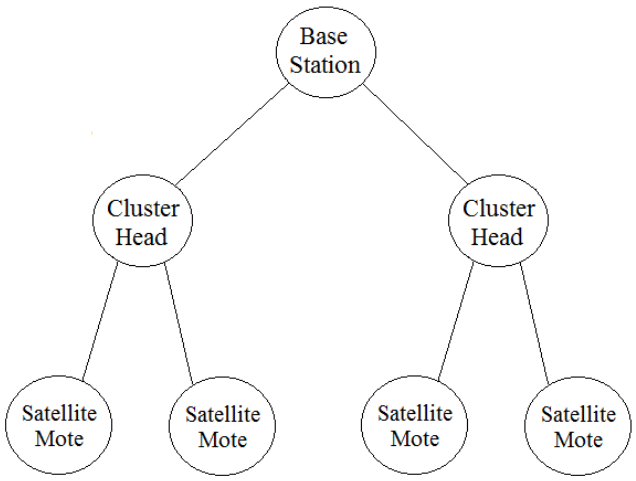

Fig. 1. Illustration of the topology of the network. In each cluster, the motes that report to the cluster head are referred to as 'satellite motes.'

IV. Damage Detection Algorithm

A. Overview

Statistical features calculated from structural vibrations can be considered random variables due to the random nature of the underlying environmental and operational factors that affect them. For our system as a whole, there will be a designated statistical feature (e.g. root-mean-square). During the execution of our algorithm, using the structural vibrations at its location, each mote will be responsible for calculating the value of the designated statistical feature. As such, each mote will have a statistical feature random variable associated with it. The statistical feature random variables in a given cluster can be expressed in vector form as in (1), where M is the number of motes in the cluster.

$$\bar{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_M \end{bmatrix} \quad (1)$$

Our algorithm consists of two primary phases: a training phase and an operational phase. During the training phase of our algorithm, each mote of the WSN will collect vibration data from the healthy structure during vibration events. For each vibration event, the mote will use the collected data to calculate the value of the designated statistical feature. Each mote will thus amass a vector of statistical feature values. When enough feature values have been gathered, the motes will transmit their values to the base station. The base station will use the data to determine which statistical features variables are correlated with each other. It will use this information to establish the clusters.

During the training phase, once the clusters are established, the base station looks at each combination of two motes in each cluster. The statistical feature variables for a given combination of two motes will be referred to as x_b and x_a. For each combination of motes in each cluster, the base station combines the statistical feature value vectors from the two motes to form a set of data points. A data point is defined to be a 2-tuple (e.g. an order list of two values) consisting of a value of the x_b statistical feature variable and value of the x_a statistical feature variable. A linear predictor, which will be denoted by $\widehat{x_{a_h}}(x_b)$, will be fit to the data points. This is appropriate since structures are designed to operate in the elastic region. The subscript h refers to the fact that the predictor is for data from the healthy structure. Linear boundary lines $b_u(x_b)$ and $b_l(x_b)$ will be established above and below, respectively, $\widehat{x_{a_h}}(x_b)$. An illustration of the predictor and boundary lines for a hypothetical set of data points can be seen in Fig. 2.

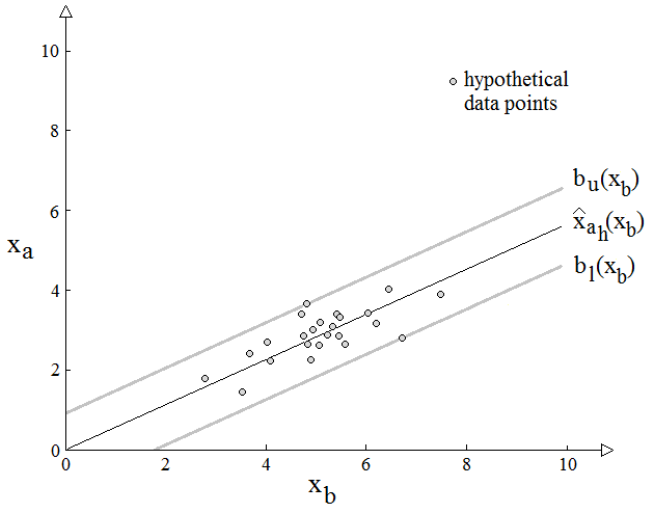

Fig. 2. Illustration of predictor and boundary lines for a hypothetical set of data points.

The boundary lines will be of the form given in (2) and (3). The constants c_u and c_l will be set such to make the boundary lines encompass the data points. To do this, for each data point, $\widehat{x_{a_h}}(x_b)$ will be evaluated at the value of x_b. The resulting value will then be subtracted from the value of x_a. Overall, then, there will be a difference value associated with each data point. For the data points lying above $\widehat{x_{a_h}}(x_b)$, the difference will be positive; for the data points that lie below $\widehat{x_{a_h}}(x_b)$, the difference will be negative. The variable c_u will be set equal to the positive difference with the greatest absolute value and c_l will be set equal to the negative difference with the greatest absolute value. The region circumscribed by $b_u(x_b)$ and $b_l(x_b)$ will be referred to as the 'bounded region.'

$$b_u(x_b) = \widehat{x_{a_h}}(x_b) + c_u \quad (2)$$

$$b_l(x_b) = \widehat{x_{a_h}}(x_b) + c_l \quad (3)$$

Each cluster will run a separate instance of the operational phase of the algorithm. The operational phase will consist of three stages: data collection, data transmission, and damage assessment. The data collection stage will run first. During this stage, each mote will collect vibration data during vibration events and use the collected data from each event to calculate the value of the designated statistical feature. In this case, each mote will amass a vector of N statistical feature values (the significance of the value of N will be discussed in Section IV-D). Once each of the motes collects N values, the data transmission stage runs. During this stage, each mote transmits its statistical feature value vector to the cluster head. The cluster head uses the statistical feature value vectors to perform the damage assessment stage. For each combination of two motes in the cluster, the cluster head combines the statistical feature value vectors from the two motes to form a set of N data points. It then checks each of the N data points to determine whether or not it falls outside the bounded region. Using this information, the cluster head determines the relative frequency of data points falling outside of the bounded region for that particular combination of motes. To do this, it will use (4). In the equation, B_n takes on the value of one if the data point falls outside the bounded region and zero if the data point falls inside the bounded region. Using (5), the relative frequency of data points falling above $b_u(x_b)$ will also be calculated. In the equation, U_n takes on the value of one if the data point falls above $b_u(x_b)$ and zero if the data points falls on or below $b_u(x_b)$. Doing this allows the cluster head to determine which way the slope of the predictor is shifting, and thus which location (x_b or x_a) the damage is closer to (this will be explained in greater detail in Section IV-D). The value of ω will then be compared to a threshold value, T; if it exceeds T, the cluster head will send an alert via inter-cluster communication to the base station informing it that damage might be present.

$$\omega = \frac{1}{N}\sum_{n=1}^{N} B_n \quad (4)$$

$$\omega_s = \sum_{n=1}^{N} U_n / \sum_{n=1}^{N} B_n \quad (5)$$

B. Data Point Location

During the damage assessment stage of the operational phase, one of the cluster heads' important tasks is the task of determining whether or not a particular data point falls outside of the bounded region. The most straightforward way to do this is to compare the data point to the boundary lines directly. Here, $b_u(x_b)$ and $b_l(x_b)$ can be evaluated at the value of x_b obtained from measurement. If the value of x_a obtained from measurement falls in between the values yielded by this evaluation, then the data point falls within the bounded region. This approach will be referred to as the 'continuous method'. A disadvantage of this method is that it could take a relatively long time to execute on low-cost microcontrollers that do not have floating point units (FPUs). Since we seek to use inexpensive microcontrollers to keep the costs of the system low, this is a relevant problem.

To speed up the process of determining the location of a data point, an approach involving look-up tables, similar to that used in [29], will be used. For each combination of motes, the x_a-x_b plane will be divided into discrete blocks and each block will be assigned to a particular location in the cluster head's memory using an injective function. To discretize the plane, the statistical feature random variables x_b and x_a will be mapped into discrete 'index value' random variables using (6). In the equation, A_{xb} and A_{xa} will determine the size of the blocks; their values will be set during the training phase, as discussed in Section IV-D. An illustration of the discretization of the x_a-x_b plane can be seen in Fig. 3. The memory location for each block will consist of two bits. One of the bits will indicate whether or not the block falls inside (wholly or partially) the bounded region. The other bit will indicate whether the block falls above $b_u(x_b)$ or below $b_l(x_b)$, in the event that it falls outside the bounded region. Note that blocks that fall either wholly or partially inside the bounded region will be referred to as 'healthy blocks.'

$$v_{xb} = \left\lfloor \frac{x_b}{A_{xb}} \right\rfloor, \quad v_{xa} = \left\lfloor \frac{x_a}{A_{xa}} \right\rfloor \tag{6}$$

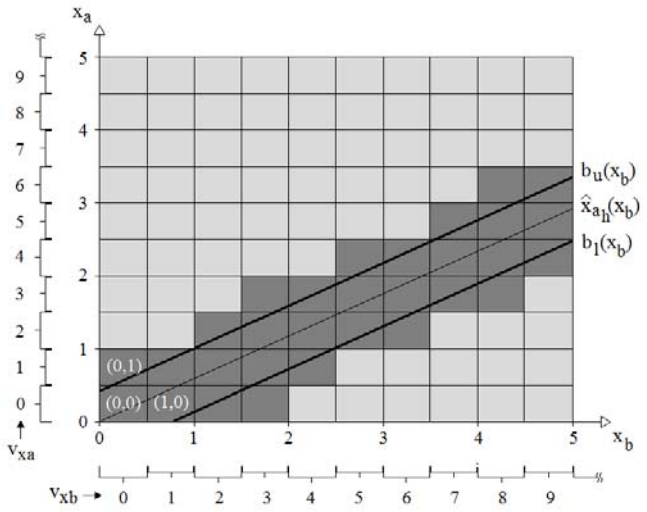

Fig. 3. Illustration of the discretization of the x_a-x_b plane, where A_{xb}=0.5 and A_{xa}=0.5. Dark grey blocks fall wholly or partially in the bounded region and light grey blocks fall outside the bounded region. Sample 2-tuples, in the format (v_{xb},v_{xa}), are shown in several of the blocks. Note that large blocks are used in the figure for illustrative purposes.

Each block will be characterized by a 2-tuple of values consisting of a value of v_{xb} and a value of v_{xa}. The values of v_{xb} and v_{xa} will be used to determine the memory address of the block. The value of v_{xb} and the value of v_{xa} will each be represented with a separate set of bits and the two sets of bits will be concatenated to form a single address. To determine the number of bits required by each index value random variable, we can define a maximum index value for each variable and use (7) and (8). The injective function used to map a 2-tuple (i.e. a block) to a memory address is shown in (9). The number of addresses required can be determined using (10). The amount of memory that will be required, in terms of bytes, can be determined by dividing the number of addresses by four (since each address is for a location consisting of two sets of bits). In determining whether or not a data point falls outside the bounded region, the values of v_{xb} and v_{xa} for the data point will be determined using (6) and the memory address of the block will be determined using (9). The address will be used to access the block's memory location to determine if the block falls outside of the bounded region; and, if it falls outside the bounded region, whether it falls above $b_u(x_b)$ or below $b_l(x_b)$. As such, using the look-up table method, the time required to determine whether or not a data point falls outside the bounded region will be constant and minimal.

Note that the look-up tables will be initialized during the training phase. For each combination of motes, once $b_u(x_b)$ and $b_l(x_b)$ are known, the base station will determine the values of A_{xb} and A_{xa} (as described in Section IV-D), discretize the x_a-x_b plane, and determine which blocks fall inside (wholly or partially) the bounded region. The blocks that fall inside the bounded region are likely representative of a healthy structure. These blocks will be considered healthy even if no data points are observed to fall into them during the training phase. This differs from [29], where a block is only considered healthy if a data point falls inside of it during the training phase. By accounting for more healthy blocks, the present method helps to decrease the likelihood of a false positive. In the event that a block falls outside the bounded region, the base station will also note whether the block falls above $b_u(x_b)$ or below $b_l(x_b)$. Once this has been done for all the combinations, the base station will transmit the information to the cluster heads to initialize their look-up tables. In particular, for each combination of motes, the base station will transmit two bits of data for each block: one bit to indicate whether or not the block falls inside (wholly or partially) the bounded region and one bit, in the event that the block falls outside the bounded region, to indicate whether the block falls above $b_u(x_b)$ or below $b_l(x_b)$.

$$w_{xb} = \lceil \log_2(v_{xb,max} + 1) \rceil \tag{7}$$

$$w_{xa} = \lceil \log_2(v_{xa,max} + 1) \rceil \tag{8}$$

$$address = v_{xb} + 2^{w_{xb}} v_{xa} \tag{9}$$

$$\#\ of\ addresses = 2^{w_{xb}+w_{xa}} \quad (10)$$

One important consideration regarding the discrete approach is that it only approximates the continuous method. This is because, in general, the boundaries of the healthy blocks will not correspond with the boundary lines. As such, blocks closest to the boundary lines will only partially fall within the bounded region. The boundary lines intersect and split these blocks. This can be seen in Fig. 3. As such, using the discrete approach, it is possible for a data point that falls outside of the bounded region to fall inside one of the healthy blocks. Given this, there is an increased possibility of a false negative. The likelihood of a false negative is related to the size of the blocks, and thus the values of A_{xb} and A_{xa}. As A_{xb} and A_{xa} are made smaller, block size decreases, and the likelihood of a false negative decreases. Overall, A_{xb} and A_{xa} should be made sufficiently small such to reduce the likelihood to an acceptable level. The process whereby A_{xb} and A_{xa} will be made sufficiently small will be discussed in Section IV-D.

C. Pseudocode

Pseudocode for the operational phase of our algorithm (which is performed by each cluster, separately) is given in Algorithm 1. Here, variables with overbars are vectors. The data collection stage will be performed first. Each mote in a given cluster will collect vibration data and calculate the value of the designated statistical feature N times. This occurs in lines 9 through 15 of the pseudocode. After N statistical feature values have been collected the data transmission stage occurs. Each mote will transmit its statistical feature value vector to the cluster head. Within each cluster, a mesh topology can be used for robustness. The data transmission stage is represented by lines 16 through 18 of the pseudocode. Once the cluster head has all of the vectors from the motes, it performs the damage assessment phase. For each combination of two motes, it determines ω and ω_s and uses this information to determine if potential damage is indicated. If damage is indicated, an alert is sent to the base station by way of inter-cluster communication. The damage assessment phase can be found in lines 19 through 43 of the pseudocode. In lines 19 and 20, the combination of motes that the cluster head operates on is determined. It should be noted that w_{xb} is determined during the training phase. Given this, the $2^{w_{xb}}$ term that appears in the address calculation in the pseudocode can be evaluated during the training phase. It appears to the cluster head simply as a constant power of two.

Algorithm 1 Algorithm for operational phase

```
1:  M ≡ number of motes in cluster
2:  i ≡ mote index: i=1,2,…,M
3:  m_i ≡ mote with index i
4:  m_1 ≡ cluster head
5:  list_i ≡ statistical feature value vector for i^th mote
6:  count_out ≡ # data points between b_u(x_b) and b_l(x_b)
7:  count_side ≡ # data points above b_u(x_b)
8:  while TRUE
9:    for n=1:N    ***data collection stage***
10:     for m_i = m_1:m_M
11:       data = collect_vibration_data();
12:       statistical_feature_value = calculate (data);
13:       list_i = append (stat_feature_value, list_i);
14:     end
15:   end
16:   for m_i = m_2:m_M    ***data transmission stage***
17:     transmit_statistical_feature_values(list_i);
18:   end
19:   for i=2:M   ***damage assessment stage***
20:     for g=1:(i-1)
21:       count_out=0;
22:       count_above=0;
23:       for n=1:N
24:         x_b = list_i(n);
25:         x_a = list_g(n);
26:         v_xb = ⌊x_b/A_xb⌋;
27:         v_xa = ⌊x_a/A_xa⌋;
28:         address = v_xb + 2^w_xb v_xa;
29:         value = get_2-bit_value (address);
30:         if value == (outside && above)
31:           count_out = count_out + 1;
32:           count_above = count_above + 1;
33:         else if value == (outside && below)
34:           count_out = count_out + 1;
35:         end
36:       end
37:       ω = count_out/N;
38:       ω_s = count_above/count_out;
39:       if ω > T
40:         send_alert_to_base_station(i,g,ω_s);
41:       end
42:     end
43:   end
44:   wait_for_stimulus();
45: end
```

From the pseudocode, it can be seen that the damage assessment stage (lines 19 through 43) requires only a handful of integer operations and two floating point operations. The integer operations consist of two divisions to calculate the index values, and a multiplication by a power of two (a shift operation) and an addition to calculate the address of the memory location corresponding to the 2-tuple of index values. Also, if the data point being processed falls outside the bounded region, the corresponding count value must be incremented. An additional incrementation is required if the data point falls above $b_u(x_b)$. The two floating point operations consist of two divisions to determine the values of ω and ω_s. Overall then, the core of our algorithm is very simple computationally. It is well suited for microcontrollers that do not contain FPUs.

D. Algorithm Background

1) Overview

In this section, the theoretical basis for our algorithm will be given. The following lemma applies to our system.

Lemma 1: For a given pair of motes, the theoretical likelihood

of a data point from a damaged structure falling outside of the bounded region tends to be greater than the theoretical likelihood of a data point falling outside the bounded region for a healthy structure.

Proof: For a given pair of motes in a given cluster, data points from the healthy structure will be distributed in a particular way. It is assumed that, for the pair of motes, boundary lines will be established (in the manner discussed in Section IV-A) that encompass the data points from the healthy structure. For the pair of motes, if a joint PDF is fit to the data points from the healthy structure, a theoretical likelihood of a data point from the healthy structure falling outside the bounded region can be determined. This likelihood is denoted by $P_{O,h}$ and can be found using (11) and (12). In (11), $p_h(x_b,x_a)$ is the joint PDF for the healthy distribution. For the pair of motes, the distribution of data points from a damaged structure will be different from the distribution of data points from the healthy structure. In particular, damage will have an effect on the slope of the predictor used for the data points [17]. This is demonstrated by our own experiment, the data from which can be seen in Section VII-B. Since damage tends to shift the distribution of data points outside of the bounded region by changing the slope of the predictor, the likelihood of a data point from a damaged structure falling outside of the bounded region tends to be greater than $P_{O,h}$.

The following theorem serves as the basis for our system.

Theorem 1: For a given pair of motes, by monitoring the relative frequency of data points falling outside of the bounded region during an operational phase, and checking to see if the relative frequency exceeds a particular threshold value, damage can be detected.

Proof: During an operational phase, for the pair of motes, data points will be gathered from the structure and the relative frequency of data points falling outside of the bounded region, ω, will be calculated. A threshold value of ω, T, will be defined. Recall that N represents the number of statistical feature values that each mote in a cluster must collect before a decision can be made regarding the condition of the structure. The likelihood of ω falling above T for a healthy structure decreases as N increases. This can be shown by treating the process of checking a data point to see whether it falls outside the bounded region as a Bernoulli trial. A sequence of independent Bernoulli trials is described by the binomial distribution. The binomial distribution for the healthy structure, in terms of ω, is given in (13). The likelihood of ω falling above T for a healthy structure (the likelihood of a false positive) will be denoted by L_{FP} and can be found using (14). From the equation it can be seen that L_{FP} decreases as N increases. This is illustrated in Fig. 4. In the figure, the binomial distribution for the healthy structure is plotted for several values of N, for a hypothetical value of $P_{O,h}$, and it can be seen that the likelihood of ω falling above T decreases as N increases. As given by Lemma 1, damage tends to increase the likelihood of data points falling outside of the bounded region, and so it will tend to produce values of ω greater than T. As such, for large values of N, a value of ω greater than T is a good indication that damage is occurring.

$$P_{I,h} = \int_{-\infty}^{\infty}\int_{-\infty}^{b_u(x_b)} p_h(x_b,x_a)\,dx_a\,dx_b - \int_{-\infty}^{\infty}\int_{-\infty}^{b_l(x_b)} p_h(x_b,x_a)\,dx_a\,dx_b \quad (11)$$

$$P_{O,h} = 1 - P_{I,h} \quad (12)$$

$$f(N\omega) = \binom{N}{N\omega} P_{O,h}{}^{N\omega}(1-P_{O,h})^{(N-N\omega)} \quad (13)$$

$$L_{FP} = \sum_{\omega=\frac{[NT]}{N}}^{1}\binom{N}{N\omega} P_{O,h}{}^{N\omega}(1-P_{O,h})^{(N-N\omega)} \quad (14)$$

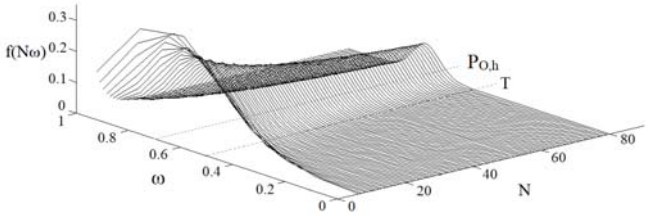

Fig. 4. Illustration of the binomial distribution for several values of N for a healthy structure (i.e. where P_O=$P_{O,h}$). A hypothetical value of $P_{O,h}$ is assumed. For any given value of N, there is the possibility that ω could fall above T (i.e. that a false positive could occur), however this likelihood diminishes as N increases.

To determine if damage is present at a particular mote in a cluster, the mote must be included in at least one of the combinations that is being monitored. To make the system robust against individual mote failure, multiple combinations that include the mote can be monitored. To do this, we monitor every possible combination of two motes in a cluster. This also makes it easier to localize damage when it occurs in between motes.

It should be noted that piecewise boundary lines can be used to encompass the training phase data points with greater detail than is possible with the boundary lines of (2) and (3); this would allow damage to be detected more quickly. We discuss such a method in [32]. In [32] we describe that, to form each boundary line, the minimum Euclidean distance between $\widehat{x_{a_h}}(x_b)$ and the boundary line can be discretely varied with the position on $\widehat{x_{a_h}}(x_b)$. To discretely vary the minimum Euclidean distance, position on $\widehat{x_{a_h}}(x_b)$ can be mapped into a set of hash keys and a minimum Euclidean distance can be assigned to each hash key for each boundary line. This method, however, requires more energy during the operational phase than the one we are now proposing. The object of our current proposal is to put forward the most energy-efficient approach possible.

2) *Determination of T and Block Size*

We desire to detect damage as close to its onset as possible. As such, we desire to make T as close to $P_{O,h}$ as possible, thus minimizing the acceptable range of ω values. As T is increased, the likelihood of a value of ω from a damaged structure falling below T (i.e. the likelihood of a false negative occurring) increases because the range of acceptable ω is larger. To make T as small as possible, maximum acceptable values of L_{FP} and N should be used (as N is made smaller for a particular L_{FP}, or

vice-versa, the value of T must be increased). Equation (14) should then be evaluated iteratively to determine the minimum value of T that can satisfy the L_{FP} and N constraints.

As discussed in Section IV-B, the usage of the look-up table method tends to inflate the likelihood of a false negative, due to the fact that the blocks only approximate the actual bounded region. This increase in likelihood can be minimized by making A_{xb} and A_{xa} sufficiently small. Since we assume that no information is available as to the distribution of data points for damage states, we cannot determine the likelihood of a false negative directly (as this would involve integrating the damage distributions over the region occupied by the healthy blocks). As such, to minimize the additional likelihood of a false negative, we will take the approach of minimizing the decrease in the likelihood of a false positive. Once the base station determines $p_h(x_b,x_a)$ for each combination of motes, for each combination, it will integrate $p_h(x_b,x_a)$ over the region occupied by the healthy blocks and treat the value as a value of $P_{I,h}$. It will then use the corresponding value of $P_{O,h}$ to determine L_{FP} using (14), and then determine the difference between the value and the actual value of L_{FP} (i.e. the value of L_{FP} associated with the region circumscribed by the boundary lines). The base station will calculate the difference for successively smaller values of A_{xb} and A_{xa} until it is negligible, or roughly two orders of magnitude less than the actual value of L_{FP}.

V. Complexity

A. Time Complexity

During the damage assessment stage of the operational phase, ω is determined for each combination of two motes in the cluster. The number of combinations of two motes in a cluster is given by $1/2M(M\text{-}1)$. For each combination, the cluster head must determine whether or not N data points fall outside of the bounded region. This is indicated by lines 23 through 36 of Algorithm 1. As such, the total number of times that the cluster head must check whether or not a data point falls outside the bounded region is equal to the total number of combinations of two motes multiplied by N. Given this, with regard to the amount of time that the cluster head must spend processing the data from the satellite motes, the algorithm grows quadratically as the number of motes M increases. As such, the time complexity of the algorithm, in M, is $O(M^2)$. Although this is a steep increase, as described in Section V-B, it can be managed by carefully choosing cluster size.

Because of the quadratic growth, the savings in execution time afforded by the usage of look-up tables over the continuous method in determining whether or not a data point falls outside the bounded region becomes significant. This can be demonstrated by using the results from the experiment discussed in Section VII-A. Here, we found the time required to determine the location of a single data point to be 16.23ms when using the continuous method and 3.29ms when using look-up tables. This translates to a speedup of roughly five. Assuming an N of 10 and an M of 10 (relatively small values) the cluster head would have to determine the location of 450 data points. For this number of data points, the continuous method would require 7.3 seconds whereas the method involving look-up tables would only require 1.5 seconds. Assuming the cluster head spends a substantial amount of time sleeping when it is not processing data from the satellite motes, this decrease represents a substantial reduction in the total power consumption of the cluster head. The savings, in terms of processing time and energy consumption, afforded by the usage of look-up tables only increase as the values of N and M are increased.

Before the damage assessment stage can run, the data transmission stage must be performed. This contributes to the total time that it takes the operational phase to run. Since there are M motes in the cluster, including the cluster head, M-1 motes will have to transmit during the data transmission stage. Each mote will have to perform $\lceil N/V \rceil$ transmissions, where V is equal to the number of statistical feature values that can fit into a single data packet. The value of V is dependent upon the particular networking specification used (e.g. ZigBee, Bluetooth low energy). The total number of transmissions that will have to occur for the damage assessment process to occur is $(M-1)\lceil N/V \rceil$. As such, with regard to transmissions the algorithm grows linearly in M ($O(M)$) and linearly in N ($O(N)$).

B. Space Complexity

During the damage assessment stage, to determine whether or not a data point falls outside the bounded region, the cluster head must store a separate look-up table for each combination of two motes. Assuming that the same values of A_{xb} and A_{xa} are used for each combination of two motes in the cluster, the total number of blocks that must be represented in a look-up table for a given combination of two motes can be found using (15). In the equation, x_{bmax} and x_{amax} are the upper bounds of the x_b and x_a statistical feature random variables defined for the mapping procedure. The total number of blocks that must be represented in the cluster head can be found by multiplying D by the total number of combinations of two motes in the cluster. From this it can be seen that, since the value of D does not depend upon M, the number of blocks that must be represented on the cluster head grows quadratically with the number of motes in the cluster. As such, the space complexity of the algorithm, in M, is $O(M^2)$. Although this is a steep increase, it can be managed by sizing the clusters appropriately. The primary consideration in the development of clusters is that the vibrations at the locations of all of the motes in the cluster be correlated with each other. If, after this consideration, the memory consumption is too great, the cluster can be split (the same is true if the processing time from Section V-A is found to be too great). Though splitting the clusters will result in a larger number of clusters in the system, system perform will not be significantly impacted. This is due to the fact that clusters generally operate independently of each other. Inter-cluster communication is rare; during the operational phase, one of the only times it will occur is when potential damage is detected and an alert has to be sent from the cluster head through the network back to the base station.

$$D = \left\lceil \frac{x_{bmax}}{A_{xb}} \right\rceil \left\lceil \frac{x_{amax}}{A_{xa}} \right\rceil \tag{15}$$

VI. Tradeoff Between Average Transmission Power and Measurement Window Size

During the operational phase, each mote collects N statistical feature values and then transmits those values to the cluster head. The transmission rate, R, in terms of transmissions per statistical feature value, per cluster is given in (16). We assume here that multi-hop communications will not be necessary, which is feasible for the distances that we plan to deploy our motes over, if each mote transmits at its maximum power.

$$R = \frac{(M-1)\lceil N/V \rceil}{N} \tag{16}$$

To a certain extent, R, and thus the average power used to transmit, can be decreased by increasing the value of N. According to (16), as N increases from N=1, R decreases up to and including N=V. At N=V+1, R jumps abruptly and then begins to decrease again up until N=2V. At N=2V+1, R jumps again and then begins to decrease again. All in all, the minima of (16) occur when N is an integer multiple of V. By evaluating (16) at multiples of V it can be shown that all of the minima have the same value. This can be seen in (17). This value is the minimum value of R that can be achieved by varying N. The maximum value of R, M-1, occurs when V equals one; this is equivalent to the case where the motes transmit their calculated statistical feature value after each data collection. Given this, R can be reduced by, at most, a factor of V as shown in (18). Overall, the ability to decrease transmission rate by increasing N represents a tradeoff. This is because we desire both a low transmission rate, for low power consumption, and a low value of N, to keep the number of measurements required to make a decision regarding the condition of the structure small. An optimum value of N can be determined by weighing the relative importance of these factors for the application at hand.

$$R(N = pV) = \frac{M-1}{V}, \; p = 0,1,2,\ldots,N \tag{17}$$

$$\frac{R_{max}}{R_{min}} = \frac{M-1}{(M-1)/V} = V \tag{18}$$

VII. Experimentation

A. Determination of Execution Time

Our method involving look-up tables has the potential to drastically reduce the time required to determine the location of a data point. We performed experiments to show the improvement in execution time that can be garnered by using look-up tables over the continuous method. In our experiments, we used a Microchip PIC18F14K50 microcontroller running at 250kHz. We first programmed the microcontroller to determine the location of a data point using look-up tables. In the program we used the method described in Section IV-B to map 2-tuples of index values to addresses. Additionally, we used the bit packing scheme described in that section to decrease the amount of memory required. To measure execution time, we raised a flag on an output pin of the microcontroller at the start of the execution of the program and lowered the flag at the end of the execution. This method produced no overhead in the code. We used a LabVIEW program to measure the duration of the flag. We found the execution time to be 3.29ms, or 206 instruction cycles. Using the energy model of (19), we also determined the energy required to execute the look-up table method. For (19), we assume a supply voltage, V_{DD}, of 3V. At this voltage, the PIC18F14K50 has a typical supply current, I_{DD}, of 550μA, assuming that it runs off of its high-frequency internal oscillator at 1MHz. Using these values, we determined the energy consumption to be 5.43μJ.

After testing the method involving look-up tables, we programmed the microcontroller to determine the location of a data point using the continuous method. The program first evaluated $b_u(x_b)$ and $b_l(x_b)$ at the value of the x_b statistical feature. It then checked to see if the value of the x_a statistical was greater than the value of $b_l(x_b)$ and less than the value of $b_u(x_b)$ (i.e. if it fell in between the two values). As with the experiment for look-up tables, execution time was determined by raising a flag on an output pin at the beginning of the execution of the program and lowering it at the end of the execution. We found the execution time to be equal to 16.23ms, or 1015 instruction cycles. Using (19), we found the associated energy consumption to be 26.8μJ. Given this, our approach involving look-up tables achieves a speedup of roughly 5 over the brute force approach involving calculations.

$$E_{execution} = V_{DD} I_{DD} t_{execution} \tag{19}$$

B. Case Study

We tested a basic version our system for one of our principal applications: bridges. In our experiment, we first executed a training phase wherein we collected data points from a structure and established $b_u(x_b)$ and $b_l(x_b)$. Then, during an operational phase, we simulated damage on the structure using a sacrificial element and collected data points again to determine if the system correctly responds to damage. In an offline fashion, the data points from the operational phase were tested to determine whether they fell inside or outside the bounded region. In this basic version of our system, each data point that fell outside the bounded region was taken to indicate damage. The relative frequency of data points falling outside the bounded region was not calculated. For our experiment, we performed measurements on the parking deck at the New Jersey Institute of Technology. We chose to experiment on a parking deck instead of a bridge because of the ease with which we could access it for experimental purposes, and because the structure of the parking deck at each level is close enough to a bridge to provide useful data.

To collect the training data, we installed sensors at two locations on one of the I-beams supporting the roadbed of one of the floors to measure acceleration at those two locations. One of the sensors was placed at the middle of the beam and the other was placed toward one of the beam's ends. We then drove an automobile several times around the floor supported by the beam. Each time, we varied the speed of the automobile. Each time the vehicle passed over the beam, we collected acceleration data from each sensor and calculated the average absolute deviation (AAD) of the first two seconds of each signal. Here, we sought to use the simplest statistical feature possible, so as to minimize the calculations that would have to be performed by the motes in our ultimate WSN system. For

each data collection, we combined the AAD values from the two acceleration signals to produce a single (x_b,x_a) data point. The resulting data can be seen in Fig. 5. In the figure, the data is labeled 'healthy'. In an offline fashion, we then used the data points to establish $b_u(x_b)$ and $b_l(x_b)$ in the manner described in Section IV-A. The boundary lines can also be seen in Fig. 5.

To determine whether the system correctly responds to damage we installed a thin sacrificial element within one of the hexagonal openings in the girder's web, close to the sensor located at the end of the beam, to simulate corrosion. We then removed the end sensor from the girder itself and fixed it to the sacrificial element. Corrosion is the type of damage that we are primarily concerned with. The sensor originally installed toward the middle of the beam was not moved. As in the training phase part of our experiment, we drove an automobile several times around the floor above, each time varying the speed of the vehicle. Each time the vehicle passed over the beam, we collected acceleration data from the two sensors and calculated the AAD of each signal. We combined the AAD values from each pass to form individual (x_b,x_a) data points. Our results are displayed as 'damage' data in Fig. 5. From the figure, it can be seen that each data point collected during the operational phase falls outside of the region bounded by $b_u(x_b)$ and $b_l(x_b)$. As such, each operational phase data point correctly indicated damage. Note that the data points from the damage case fall far outside the region bounded by $b_u(x_b)$ and $b_l(x_b)$. This tends to indicate that the method will be useful for detecting damage that is much more subtle than the damage simulated (through the use of the sacrificial element) in the experiment.

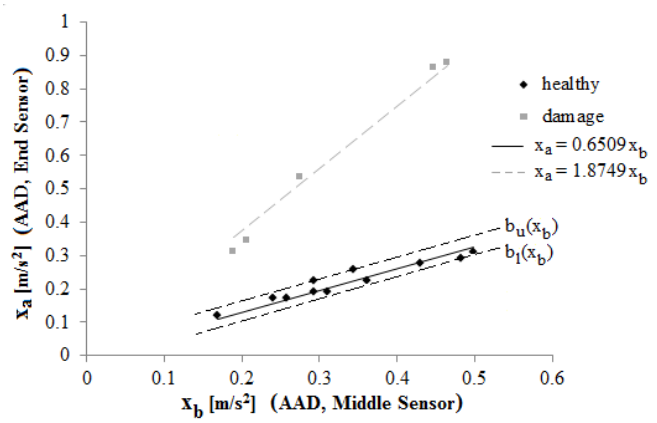

Fig. 5. Data from the healthy and damaged structures.

VIII. Conclusion

Overall, we have proposed a novel technique for assessing structural condition using WSNs. Like several earlier methods, our technique uses a statistical approach. Unlike many other statistical approaches, however, our technique is very efficient computationally. By employing an innovative technique involving look-up tables we were able to achieve an implementation involving only 4-6 integer operations and 2 floating point operations. We benchmarked our look-up table approach against the brute force technique involving calculations by measuring the execution time of each implementation on a low-power Microchip microcontroller. From this, we determined that the look-up table approach yields a speedup of roughly 5. In addition, we investigated a tradeoff between average transmission power and the size of the measurement window. Here, we found that the transmission rate can be decreased by a factor equal to the data payload size of the networking specification used, at the expense of a longer measurement window. Finally, we benchmarked our technique to verify its efficacy in the case of bridges. Future work should include investigating the power consumption of our system more thoroughly. In particular, the savings in power consumption garnered by using our look-up table approach as a percentage of total power consumption should be determined. The feasibility of powering the system off of vibrations should also be investigated.

References

[1] M. J. Whelan, M. V. Gangone, K. D. Janoyan, and R. Jha, "Real-time wireless vibration monitoring for operational modal analysis of an integral abutment highway bridge," *Eng. Structures*, vol. 31, no. 10, pp. 2224-2235, Oct. 2009.

[2] M. Ceriotti, L. Mottola, G. P. Picco, A. L. Murphy, S. Guna, M. Corra, et al., "Monitoring heritage buildings with wireless sensor networks: The Torre Aquila deployment," in *Int. Conf. on Inform. Process. in Sensor Networks*, Apr. 2009, pp. 277-288.

[3] K. Sukun, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, et al., "Health monitoring of civil infrastructures using wireless sensor networks," in *6th Int. Symp. on Inform. Process. in Sensor Networks*, Apr. 2007, pp. 254-263.

[4] M. Bocca, L. M. Eriksson, A. Mahmood, R. Jäntti, and J. Kullaa, "A synchronized wireless sensor network for experimental modal analysis in structural health monitoring," *Comput.-Aided Civil and Infrastructure Eng.*, vol. 26, no.7, pp. 483-499, Oct. 2011.

[5] G. Hackmann, F. Sun, N. Castaneda, C. Lu, and S. Dyke, "A holistic approach to decentralized structural damage localization using wireless sensor networks," *Comput. Commun.*, vol. 36, no.7, pp. 29-41, Dec. 2012.

[6] M. Bocca, J. Toivola, L. M. Eriksson, J. Hollmen, and H. Koivo, "Structural health monitoring in wireless sensor networks by the embedded Goertzel algorithm," in *IEEE/ACM Int. Conf. on Cyber-Physical Syst. (ICCPS)*, Apr. 2011, pp. 206-214.

[7] S. Chesné and A. Deraemaeker, "Damage localization using transmissibility functions: a critical review," *Mech. Syst. and Signal Process.*, vol. 38, no. 2, pp. 569-584, July 2013.

[8] B. Phares, P. Lu, T. Wipf, L. Greimann, and J. Seo, "Field validation of a statistical-based bridge damage-detection algorithm," *J. of Bridge Eng.*, vol. 18, no. 11, pp. 1227-1238, Nov. 2013.

[9] M. J. Daigle and K. Goebel, "Model-based prognostics with concurrent damage progression processes," *IEEE Trans. on Syst., Man, and Cybern.: Syst.*, vol. 43, no. 3, pp. 535-546, May 2013.

[10] V. Lamsa and T. Raiko, "Novelty detection by nonlinear factor analysis for structural health monitoring," in *IEEE Intern. Workshop on Mach. Learning for Signal Process. (MLSP)*, Aug.-Sept. 2010, pp. 468-473.

[11] A. Deraemaeker, E. Reynders, G. De Roeck, and J. Kullaa, "Vibration-based structural health monitoring using output-only measurements under changing environment," *Mech. Syst. and Signal Process.*, vol. 22, no. 1, pp. 34-56, Jan. 2008.

[12] Z. Hongyang, G. Junqi, X. Xiaobo, B. Rongfang, and S. Yunchuan, "Environmental effect removal based structural health monitoring in the Internet of Things," in *Seventh Int. Conf. on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS)*, July 2013, pp. 512-517.

[13] W.-H. Hu, E. Caetano, and Á. Cunha, "Structural health monitoring of a stress-ribbon footbridge," *Eng. Structures*, vol. 57, no. 0, pp. 578-593, Dec. 2013.

[14] J. Kullaa, "Structural health monitoring under nonlinear environmental or operational influences," *Shock and Vibration*, p. 9, May 2014.

[15] A. G. González and S. D. Fassois, "Vibration-based statistical damage detection for scale wind turbine blades under varying environmental conditions," in *Surveillance 7*, Chartres, France, 2013.

[16] S.-S. Jin and H.-J. Jung, "Vibration-based structural health monitoring using adaptive statistical method under varying environmental condition," in *Proc. of SPIE Vol. 9064*, Mar. 2014.
[17] N. H. M. Kamrujjaman Serker, Z. Wu, and S. Li, "A nonphysics-based approach for vibration-based structural health monitoring under changing environmental conditions," *Structural Health Monitoring*, vol. 9, no. 2, pp. 145-158, 2010.
[18] D. Gorinevsky and G. Gordon, "Spatio-temporal filter for structural health monitoring," in *Amer. Control Conf.*, June 2006.
[19] A. D. Kenyon, V. M. Catterson, S. D. J. McArthur, and J. Twiddle, "An agent-based implementation of hidden Markov models for gas turbine condition monitoring," *IEEE Trans. on Syst., Man, and Cybern.: Syst.*, vol. 44, no. 2, pp. 186-195, Feb. 2014.
[20] A. Scianna, Z. Jiang, R. Christenson, and J. DeWolf, "Implementation of a probabilistic structural health monitoring method on a highway bridge," *Advances in Civil Eng.*, vol. 2012, 2012.
[21] J. Peter Lynch, A. Sundararajan, K. H. Law, A. S. Kiremidjian, and E. Carryer, "Embedding damage detection algorithms in a wireless sensing unit for operational power efficiency," *Smart Materials and Structures*, vol. 13, no. 4, Aug. 2004.
[22] Q. W. Zhang, "Statistical damage identification for bridges using ambient vibration data," *Comput. & Structures*, vol. 85, no. 7-8, pp. 476-485, Apr. 2007.
[23] E. Hidalgo et al., "Wireless structural health monitoring system based on autoregressive models," in *38th Annu. Conf. on IEEE Ind. Electron. Soc.*, Oct. 2012, pp. 6035-6040.
[24] O. Alexander and G. Spielberger, "Comparison of AR and ARMA processes in structural health monitoring," *Proc. in Appl. Mathematics and Mechanics*, vol. 14, no. 1, pp. 765-766, Dec. 2014.
[25] M. M. Alves et al., "Damage prediction for wind turbines using wireless sensor and actuator networks," *J. of Network and Comput. Appl.*, vol. 80, pp. 123-140, Feb. 2017.
[26] S. Hoell and P. Omenzetter, "Optimal selection of autoregressive model coefficients for early damage detectability with an application to wind turbine blades," *Mech. Syst. and Signal Process.*, vol. 70–71, pp. 557-577, Mar. 2016.
[27] C. W. Follen, M. Sanayei, B. R. Brenner, and R. M. Vogel, "Statistical bridge signatures," *J. of Bridge Eng.*, vol. 19, no. 7, July 2014.
[28] P. Henriquez, J. B. Alonso, M. A. Ferrer, and C. M. Travieso, "Review of automatic fault diagnosis systems using audio and vibration signals," *IEEE Trans. on Syst., Man, and Cybern.*: Syst., vol. 44, no. 5, pp. 642-652, May 2014.
[29] W. Contreras and S. Ziavras, "Wireless sensor network-based pattern matching technique for the circumvention of environmental and stimuli-related variability in structural health monitoring," *IET Wireless Sensor Syst.*, vol. 6, no. 1, pp. 26-33, Feb. 2016.
[30] R. K. Shakya, Y. N. Singh, and N. K. Verma, "Generic correlation model for wireless sensor network applications," *IET Wireless Sensor Syst.*, vol. 3, no. 4, pp. 266-276, Dec. 2013.
[31] D. de Jager and J. S. Reeve, "Efficient information valuation and costing for distributed wireless sensor networks," *IET Wireless Sensor Syst.*, vol. 2, no. 3, pp. 191-200, Sept. 2012.
[32] W. Contreras and S. Ziavras, "Wireless sensor network-based infrastructure damage detection constrained by energy consumption," presented at the *7th IEEE Annu. Ubiquitous Computing, Electron. & Mobile Commun. Conf.*, Oct. 2016.