Cyber-Physical Codesign of Distributed Structural Health Monitoring with Wireless Sensor Networks

Gregory Hackmann, Weijun Guo, Guirong Yan, Zhuoxiong Sun, Chenyang Lu, and Shirley Dyke

Abstract—Our deteriorating civil infrastructure faces the critical challenge of long-term structural health monitoring for damage detection and localization. In contrast to existing research that often separates the designs of wireless sensor networks and structural engineering algorithms, this paper proposes a cyber-physical co-design approach to structural health monitoring based on wireless sensor networks. Our approach closely integrates (1) flexibility-based damage localization methods that allow a tradeoff between the number of sensors and the resolution of damage localization, and (2) an energy-efficient, multi-level computing architecture specifically designed to leverage the multi-resolution feature of the flexibility-based approach. The proposed approach has been implemented on the Intel Imote2 platform. Experiments on a simulated truss structure and a real, full-scale truss structure demonstrate the system's efficacy in damage localization and energy efficiency.

Index Terms—wireless sensor networks, structural health monitoring, cyber-physical systems

1 Introduction

Deterioration of civil infrastructure is a growing problem in the US and around the world. For example, during their lifetimes, bridges face environmental corrosion, persistent traffic and wind loading, extreme earthquake events, material aging, etc., which inevitably result in structural deficiencies. According to the American Society for Civil Engineers 2009 Report Card for America's Infrastructure, "more than 26%, or one in four, of the nation's bridges are either structurally deficient or functionally obsolete" [5]. Due to the expense of retrofitting a wired sensor infrastructure, most of these structures are not currently being continuously monitored.

Recent years have seen growing interest in SHM based on wireless sensor networks (WSNs) due to their low installation and maintenance expenses. WSNs permit a dense deployment of measurement points on an existing structure, facilitating accurate and fault-tolerant damage identification techniques without installing a fixed wired infrastructure [27]. Indeed, numerous SHM systems have been proposed in literature which leverage WSNs to collect raw sensor data [9], [10], [19], [37]. These systems are generally designed to support traditional centralized SHM methods, with special consideration to the limited bandwidth and energy supplies that are not present under a traditional system of wired sensors.

However, systems which treat WSNs as simple data collection devices for centralized SHM methods inherently suffer from high energy consumption and prolonged detection latencies. For example, a state-of-art system deployed at the Golden Gate Bridge required 9 hours to collect a single round of data from 64 sensors, resulting in a system lifetime of 10 weeks from four 6V lantern batteries [29]. The high latency and relatively short lifetime arose from the underlying SHM method being designed separately from the WSN system. The SHM method required the WSN to reliably deliver the entire raw sensor dataset to the base station for centralized processing, inherently placing a high network burden on the WSN system.

What is needed is a fundamentally different *cyber-physical* approach which considers both the constraints of the underlying WSN system (the *cyber* components) and the SHM requirements (the *physical* components) in its numerical approach. This can be achieved by leveraging the powerful processing capability of wireless sensor "motes" to partially process locally-collected data, extracting (and exchanging) only features relevant for SHM. Recent studies demonstrate the potential for distributed SHM approaches to significantly reduce energy cost through localized data processing [7], [17], [27], [41].

In this paper, we present a hierarchical decentralized SHM system that implements *flexibility-based* damage identification and localization. Flexibility-based methods accurately identify and localize damage on a wider range of structures than previous decentralized algorithms like DLAC [25], by explicitly correlating data across multiple sensors. Our hierarchical system organizes nodes into clusters using a novel *multi-level search* approach that incrementally activates sensors in the damaged regions, allowing much of the network to remain asleep. By leverage the Intel Imote2 [13] platform's computational power to perform in-network processing, nodes save energy and bandwidth by only transmitting the intermediate results related to the flexibility calculation.

In this paper, we make the following contributions. (1) We propose a *cyber-physical architecture* which effi-

ciently maps flexibility-based damage identification and localization methods onto a distributed WSN. (2) We describe an implementation of this architecture on top of the TinyOS operating system [1] and ISHM services toolsuite [2]. (3) We evaluate this implementation on a simulated truss structure and a real, full-scale truss structure, successfully localizing damage on both structures to the resolution of a single element. Latency and power consumption data collected during these experiments demonstrate the efficiency of our approach.

The remainder of this paper is organized as follows. Section 2 describes related SHM systems in literature. In Section 3, we discuss the basic numerical methods used by our flexibility-based damage localization. Section 4 presents our mapping of these methods into an efficient distributed architecture. Section 5 describes our implementation of this distributed architecture on top of the Intel Imote2 platform. Section 6 provides an empirical evaluation of our system, demonstrating that it can efficiently localize damage to two representative structures. Finally, we conclude in Section 7.

2 Related Work

A UC Berkeley project to monitor the Golden Gate Bridge [19] is one of the first large-scale deployments of smart sensor networks for SHM purposes. Vibration data is collected and aggregated at a base station under a centralized network architecture, where frequency domain analysis is used to perform modal content extraction. However, it took nearly a full day to transmit sufficient data for such computations. Researchers at Clarkson University have implemented a wireless sensor system for modal identification of a full-scale bridge structure in New York [15]. Both modal identification and quantification of static responses are performed using a centralized network architecture. Wisden [37] provides services for reliable multi-hop transmission of raw sensor data, using run-length encoding to compress the data before transmission. These centralized approaches suffer from two fundamental limitations. First, data may only be collected from a limited number of nodes in a reasonable time frame, which would allow the system to only detect the most severe (and probably visually apparent) damages. Second, such systems are inadequate for timely detection of structural failures resulting from extreme events (e.g., earthquakes) due to the prolonged time needed for collecting and analyzing data.

BriMon [10] partially addresses the communication bottleneck by sampling data at 400 Hz and averaging this data over 40 Hz windows. The data resolution and network size (a maximum of 12 nodes per span) supported by BriMon may not be fine-grained enough for damage detection and localization on complex structures. A deployment in the Torre Aquila heritage building [9] uses lossless compression to deliver heterogeneous sensor data to a sink node. The network burden was eased by requiring only three acceleration sensors, plus 1–10 environmental and deformation readings every 10 minutes, to monitor the specific structure's health. p-SPEM [21] uses a strategy of placing sensors to reduce networking overhead, and has been shown to significantly increase system lifetime in simulated deployment on the Ting Kau Bridge and real deployment on the Guangzhou New TV Tower. EleSense [35] similarly aims to reduce networking costs by using the Guangzhou New TV Tower's elevators as mobile base stations. These strategies specifically target the networking cost, an important but incomplete part of the energy budget; our distributed architecture aims to also reduce the significant sensing and computation costs.

These limitations motivate a *co-design* approach which addresses both SHM and WSN concerns in a holistic manner. An integral part of such a solution is the adoption of distributed SHM solutions [23], [34]. Researchers at the University of Illinois at Urbana-Champaign have experimentally validated an SHM system that employs a smart sensor network deployed on a scale three-dimensional truss model [27], [33]. Results demonstrate that the adopted SHM system is effective for damage identification and localization; however, significant communication is involved to cross-correlate data, resulting in significant energy consumption.

Lynch et al. [36] implemented a low-cost and rapid-to-deploy wireless structural monitoring system on a long-span cable-stayed bridge in Taiwan. The full-scale test was conducted by collecting ambient vibration data of the bridge and analyzing it in situ by two modal identification methodologies, the stochastic subspace identification method (SSI) and frequency domain decomposition method (FDD). Modal ID results led to the determination of a total of 10 modal frequencies and corresponding mode shapes within a frequency range of 0–7 Hz. Lynch et al. [40] also implemented an automated modal identification by optimizing output-only modal methods (FDD with peak-picking) for a distributed wireless sensor network. The distributed implementation, tested in a balcony of a theater, used a parallel data processing and reduced communication scheme to ensure scalability and power efficiency in the WSN. Their implementation proposes three network topologies to yield a two-node based data sharing chain. This implies the partial mode shape identified from each pair of nodes has to be recombined to recreate the complete mode shape necessary for damage detection. This strategy would potentially amplify the recombination error, if any one of the sensor nodes is unreliable. Jindal and Liu [18] employ singular value decomposition (SVD) to extract a structure's mode shapes and propose a clustering scheme to distribute this heavyweight computation among the network. Our distributed architecture uses a simpler peak-picking approach to reduce the size of the SVD computation, making it feasible to perform on a single cluster head.

In our own prior work, we designed and experimentally validated a distributed approach based on the Damage Location Assurance Criterion (DLAC) method [7],

[17]. However, DLAC has several intrinsic limitations in its SHM capabilities. First, the user must pre-specify the damage patterns to identify and localize. Second, DLAC is not sensitive to small damages in a structure because it only monitors the structure's natural frequencies, and because it does not correlate readings across sensors. Finally, DLAC can only properly localize damage to asymmetric structures. These limitations occur because there is effectively no collaboration among sensors: each sensor's readings are handled independently, and are only combined at the very end to compensate for node failures and sensor noise. Alleviating these limitations requires a fundamentally new architecture which leverages collaboration among sensors to enhance the damage identification and localization results.

This paper focuses on the architecture and case study on a real physical structure. Our distributed architecture assumes a clustering scheme is available to designate clusters and cluster heads under simple constraints on node placement (discussed in Section 4.3). From a networking perspective, clustering is a challenging but well studied problem in distributed networking which is complementary to the work described in this paper. Liu et al. [22] propose two centralized and one decentralized clustering scheme for SHM applications which use distributed modal analysis. Clusters generated by these schemes meet the requirements of this class of SHM applications: all nodes are included in at least one cluster and connected to cluster heads with single-hop links; minimum cluster size is enforced; and all clusters are connected via overlapping nodes. designed around the requirements of distributed modal analysis. Using their SVD-based SHM scheme as a motivating example, Jindal and Liu [18] propose a general, near-optimal distributed clustering algorithm for WSNs. A survey of other relevant clustering methods may be found in [4].

3 Damage Localization Approach

In this section, we introduce the *physical* (structural engineering) aspects of our decentralized damage localization system. Our system is based on the *flexibility based* family of damage localization algorithms. The intuition behind these methods is that structures will flex slightly when a force is applied, as shown in Figure 1. As a structure weakens, its stiffness decreases, and thus its flexibility changes. Changes the structure's flexibility over its lifetime can be used to identify and localize damage [30]. We have chosen this family of methods because they address the aforementioned limitations in DLAC. Moreover, as we discuss in Section 4, they enable us to develop a multi-level system architecture specifically optimized for this approach.

While flexibility-based methods are well-known in structural engineering literature, the existing research generally deals with algorithmic issues (i.e., selecting the best numerical methods for damage identification and localization) rather than efficiently deploying these methods on a distributed architecture for WSNs. We will focus here on the details of these algorithms that are most relevant to our system design; more mathematical details can be found in [14], [38].

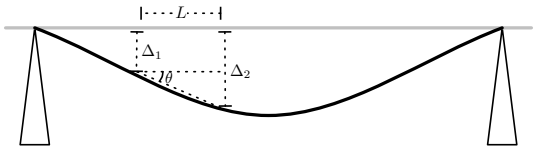


Fig. 1. Structural deflection

Flexibility-based methods are executed in two stages. When the system is first turned on, a baseline structural modal identification is performed. The sensors simultaneously collect vibration data. Multiple sensors' data are correlated to identify the structure's *modal parameters* (natural frequencies and mode shapes). The modal parameters are then further processed to compute the structure's *flexibility matrix*.

Online, the data collection and processing phases above are repeated, and the base station produces a new flexibility matrix. By subtracting the new matrix from the stored one, the base station can determine if the structure is damaged (and if so, identify the damaged region).

Figure 2 illustrates the main components of flexibility-based methods. The structure's modal parameters are identified using Frequency Domain Decomposition (FDD), an existing structural engineering technique which can be decomposed into several stages. Traditionally, FDD is executed as follows. (1) All nodes in a cluster simultaneously collect D vibration samples using their onboard accelerometers. D's size depends on structural properties (like its complexity and material) as well as the modes we are interested in, and is typically hundreds or thousands of samples. (2–3) Each node independently performs a Fast Fourier transform (FFT) and power spectrum analysis on the vibration data, transforming it into magnitudes in the frequency domain. (4) D magnitudes collected from each node are correlated to compute a Cross Spectral Density (CSD) matrix. (5) A Singular Value Decomposition (SVD) is performed on the CSD matrix at each of D discrete frequencies. The singular value in each matrix is collected to form a vector, and the structure's P lowest natural frequencies are identified as the peaks in this vector. The corresponding mode shapes can be estimated from the first column of the corresponding left SVD matrix.

The results are input into a flexibility-based method. Our system uses two specific methods: the Angles-Between-String-and-Horizon flexibility-based method (ASHFM) [14] and the Axial Strain flexibility-based method (ASFM) [38]. These two methods can localize damage to a specific element on beam-like and truss-like structures, respectively. A discussion of these methods may be found in Appendix A.

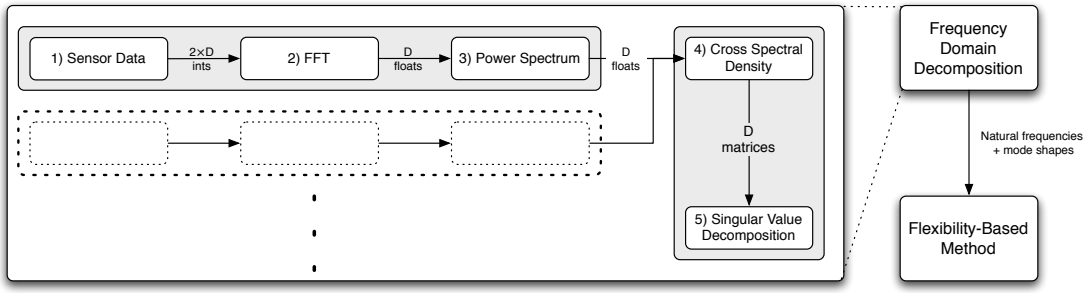


Fig. 2. The data flow of a traditional flexibility-based method

4 Distributed Architecture

The numerical methods discussed above have been designed with centralized networks in mind, where sensors are used as simple data collection devices that can stream large data sets to a central server over a wired backbone. Under a WSN, this approach is inappropriate because of the nodes' limited network and energy resources. However, in order to design an efficient decentralized architecture, we can leverage a particularly powerful feature of these flexibility-based methods. Specifically, they enable a tradeoff between energy consumption and localization resolution: the more nodes that are activated, the finer-grained the damage localization.

We leverage this feature to construct an energy-efficient, *multi-level* damage localization system which selectively activates additional sensors at each level in order to more precisely localize structural damage. In the common case that the structure is undamaged, only a minimal subset of nodes are enabled, considerably reducing the system's energy and bandwidth consumption. This approach naturally maps to a hierarchical, cluster-based distributed network architecture. To promote a more efficient mapping onto our distributed system, we also leverage an existing *peak picking* technique to reduce the data flow among participating sensors.

4.1 Multi-Level Damage Localization

Although adding more sensors can improve a flexibility-based method's localization results, only a handful of sensors are needed to accurately *identify* damage. In the first stage of the multi-level search, this minimal number of sensors are enabled as a single cluster. Damage identification and localization is performed using this single cluster. In the common case that no damage is identified, the search ends and all the nodes return to sleep.

When damage is identified, the flexibility-based method will also output coarse-grained damage localization. For example, ASHFM will identify two adjacent sensors surrounding each damage location on the structure. In the next round of the multi-level search, the system activates additional sensors in the region of interest and repeats the entire procedure. This second round subsequently localizes the damage to a smaller region than the first round. The system may repeat this drill-down procedure to achieve even finer grained results until the desired resolution is reached.

The key feature of this approach is that it does not activate the entire sensor network at once. Moreover, participating nodes do not run continuously and thus are not awake for the entire lifetime of the application. Instead, relatively few sensors perform the first round of damage detection on a predetermined schedule. When damage is identified, only sensors in the area of interest are added to the search. All nodes spend the time in-between rounds sleeping. On-demand damage detection (e.g., after an earthquake) may be supported with a duty-cycling MAC layer like BoX-MAC-2 [26] in exchange for a small energy overhead. As a result, many nodes are able to remain asleep for part or all of the multi-level search, and all nodes will sleep for a substantial portion of the deployment's lifetime. As we discuss in Section 5.2, the reduced energy burden can also be distributed across the network by activating different subsets of the network at different times. This approach also scales to larger structures, since the cost of the search is no longer proportional to the size of the structure.

4.2 Network Hierarchy

Once participating nodes are selected, they are each assigned one of three roles: *cluster member*, *cluster head*, and *base station*. A node's role determines what data it handles as well as its level in the network hierarchy.

Based on these roles, the system operates as follows. The cluster members collect raw vibration samples from their onboard accelerometers. They then carry out an FFT to transform the vibration response into frequency domain data, followed by a power spectrum analysis.

The cluster head nodes aggregate the extracted power spectrum data from their cluster members. There, the CSD and SVD are carried out to extract the structure's mode shape vector. The cluster heads then transmit the mode shapes to a single base station node, which calculates the structure's flexibility. The flexibility is then used to identify and localize any structural damage.

Each cluster acts as an independent unit, with the cluster head coordinating nodes within its cluster and ultimately transmitting its (relatively small) mode shape data to the base station for final processing. Using multiple clusters can reduce the network burden and spread

the computational cost across the cluster heads, without affecting the final output of damage localization.

4.3 Node Selection and Clustering

The flexibility-based methods used in our architecture have relatively simple constraints on node selection for damage detection performance. Hence, there is significant flexibility in choosing nodes that participate at each stage of the multi-level search. We note that our current implementation treats the selection and clustering of nodes as an orthogonal problem. However, we recognize that the clustering scheme is an important component of a fully-automated, large-scale SHM deployment. We outline here a general approach that may be used to efficiently cluster nodes within our architecture. Such an approach may be implemented using existing clustering algorithms in literature, like those discussed in Section 2.

In the class of structures considered in this paper, truss-like structures, one node for each bay of the structure is typically required to perform the first round of damage localization at sufficient accuracy [8]. The system's lifetime may be extended by rotating among the nodes in each bay based on network conditions or each node's residual energy. By activating one node per bay, the level-1 results will correctly detect damage and provide a coarse-grained damage localization.

Once damage is detected, there are various ways to "drill down" the search to the desired localization accuracy. For example, given an affected area with a large number of sensors, all these sensors may be activated for a single additional round of damage localzation. Alternatively, a small subset of these sensors may be activated for the second round to achieve medium-granularity localization; more sensors in the smaller affected area may be activated for a third round to achieve even finer-grained localization; etc. Though both approaches will ultimately achieve the same damage localization result, they can have very different energy footprints. The former approach activates sensors that would not otherwise need to be activated, while the latter approach may cause some sensors to be activated for multiple levels of the search. The costs of each strategy may be estimated at run-time based on energy measurements collected beforehand (such as those in Section 6.1.1), and then account for the energy consumption and residual energy to choose an optimized clustering strategy using a clustering algorithm in literature.

Likewise, partitioning the network into clusters and selecting cluster heads will not affect the final output, but can significantly affect the system's energy budget. These selections directly affect the networking cost of all participating nodes, which must deliver data to their designated cluster head. Moreover, a node acting as cluster head will have significantly increased network and computational burdens as shown in Section 6.1.1. This burden may be alleviated by rotating the cluster head among the network over the application lifetime. Again, clustering schemes like those discussed in Section 2 can be employed to partition the network and select cluster heads in order to increase the network's lifetime.

4.4 Enhanced FDD

Efficiently implementing this architecture for a flexibility-based system is challenging because there are no obviously "best" places to introduce network communication: CSD and SVD are necessarily computed on a single node with access to all the other cluster members' data, and the prior steps all have very large outputs (hundreds or thousands of points). In order to achieve truly energy-efficient behavior, we must optimize the FDD algorithm's data flow to promote an efficient mapping onto wireless sensor networks.

We leverage an optimization proposed in [36], [41] that adds a *peak picking* stage to FDD. To illustrate this optimization, we note that most of the computations in the FDD routine do not contribute to the final results. As described in Section 3, the CSD step normally requires the cluster head to pool D data points from each of its cluster members. This data is processed into D CSD matrices, which the SVD routine further processes into D outputs and discards all but the P corresponding to the structure's natural frequencies (note that $P \ll D$). A key observation about this procedure is that the ith CSD matrix is only constructed using the ith power spectrum data point from each cluster member. Moreover, only the P CSD matrices corresponding to the structural's natural frequencies contribute to the FDD stage's final output.

The peak picking routine allows each node to independently identify these P natural frequencies solely from local data. Hence, only those P relevant data points are passed onto the CSD stage, which in turn passes only the relevant P matrices onto the SVD stage. In this way, both the computational and communication cost of identifying modal parameters are reduced considerably. We emphasize that the data which the nodes withheld would not have contributed to the final flexibility computation. Hence, even though significantly fewer data are transmitted and processed, there is no loss in damage identification or localization performance.

5 Implementation

We have built a proof-of-concept implementation of our system on top of the Imote2 [13] sensor platform using the TinyOS operating system [1]. This implementation utilizes the ISHMP services toolsuite [2] developed by the Illinois Structural Health Monitoring Project, which provides subsystems for sensor data acquisition, reliable data transmission, remote procedure calls, and time synchronization based on the FTSP protocol [24].

5.1 Hardware Platform

The Imote2 is an advanced wireless sensor node platform built around the low-power PXA271 XScale processor and 802.15.4-compliant radio hardware (Chipcon

CC2420) with a built-in 2.4GHz antenna. While our proposed approach to SHM is not inherently tied to a particular platform, the Imote2 offers several salient improvements over previous generation WSN platforms that are particularly useful for our application.

First, the PXA271 CPU has 256 KB of embedded SRAM and can address 32 MB of on-board SDRAM, providing copious space for computations. In contrast, platforms such as the TelosB [31] and MICAz [12] have access to only 4–10 KB of RAM, which would not even be enough to store the entire raw sensor reading dataset. Such platforms would either be restricted to purely streaming computations or would have to swap data in and out of onboard flash, a potentially expensive operation.

Second, the PXA271 CPU can be dynamically clocked from 13–416 MHz, allowing nodes to increase CPU speed when needed (e.g., while collecting high-resolution sensor data) and decrease speed at other times to save energy. Third, the Imote2 is a modular stackable platform which can be expanded with extension boards to customize the system to a specific application. The SHM-A [32] sensor board provides an add-on accelerometer which we have confirmed to be sufficiently accurate for our SHM application. Fourth, the Imote2 is equipped with 32 MB of flash memory, which allows us to deploy the entire application on all nodes in the network. We take advantage of this capability to dynamically reconfigure the network without having to re-flash the nodes with new software, as discussed below.

5.2 Software Platform

As described above, the system is implemented in the nesC programming language on top of the TinyOS 1.1 operating system. Several major components from UIUC's ISHMP Toolsuite 3.0 were leveraged to ease implementation. Specifically, ISHMP's `RemoteCommand` RPC subsystem is used to reliably coordinate motes and collect partial results, as discussed below; and the `SensingUnit` components are used to start data collection simultaneously across all the participating motes.

At the start of the procedure, the base station constructs a configuration message containing information about the cluster division and the assignment of roles within each cluster. The base station disseminates this packet to the cluster head, which in turn disseminates it to the other cluster members. Because the Imote2 platform is equipped with copious flash memory, all nodes are programmed with the code for all roles. Thus, nodes can handle the reconfiguration message by simply changing their configuration parameters in RAM.

After all nodes are configured, the base station disseminates a control message into the network to start data collection; this message propagates to the cluster heads and cluster members in a similar fashion to the configuration packet. After completing data collection and computation, cluster members deliver the partial results to the cluster heads. Similarly, the cluster heads deliver mode shape data to the base station after completing their computations. Rather than implementing the complex final damage calculations directly on the base station, the base station outputs the cluster heads' data over its serial port. This output is collected at a PC attached to the mote, and the final computations are performed in MATLAB. If damage is identified, more fine-grained damage localization may be triggered as discussed in Section 4.

Our architecture uses ISHMP's `RemoteCommand` RPC subsystem as its communication primitive. Sensor nodes expose remote calls for configuration and performing damage detection; partial results are provided as return values to these calls. For example, the base station configures the cluster heads by calling `ManagerConfig`, which in turn configures the cluster's members by remotely calling `LeafConfig`. `RemoteCommand` provides built-in retransmission of lost packets and optional multi-hop routing, key features for SHM applications.

As discussed in [8] and demonstrated in our experimental results, one sensor per bay typically leads to accurate level-1 damage detection. To enhance localization accuracy, additional nodes are woken up only in the area of interest. We note that the current implementation does not automate this node selection process. Moreover, the base station does not currently allow users to configure networks containing more than one cluster. These are not fundamental limitations of the architecture, and could be lifted with additional base station code implementing existing clustering techniques in literature (see Section 2).

5.3 Development Process

One of the most significant challenges faced in implementing a distributed SHM system is that it currently requires expertise in two specialized domains: structural engineering *and* system-level programming. The implementation challenge is compounded by the embedded, low-power hardware used in these deployments. Standard development techniques like source-level debugging are often taken for granted in PC software development, but on embedded platforms are typically only available with specialized debugging hardware (if at all) that is impractical to deploy in the field.

For these reasons, we found that it was critical to establish a multi-phase development process involving both the structural engineers and the computer scientists. This process was motivated by our experiences developing a scrapped prototype. Specifically, we found that reflashing devices became a bottleneck in debugging, and that memory bugs were both the most numerous class of bugs and the most difficult to debug in situ. Our experiences debugging this ad-hoc prototype led to the difficult decision to re-implement from scratch under a more controlled development process. This process involved four stages, described below. By progressing from a MATLAB prototype to a full embedded implementation, we were better able to divide the process up by specialized skills

and perform unit-testing before deployment time. More generally, we believe that this process provides a good set of guidelines for developing complex cyber-physical systems involving multiple specialized disciplines.

5.3.1 MATLAB Prototype

We first prototyped the numerical method in the MATLAB environment. Although MATLAB code cannot be directly deployed on most embedded platforms, the high-level nature of the language makes it a good environment for structural engineers to test and fine-tune the underlying mathematical approach.

MATLAB was also used to generate simulated sensor traces based on a truss's numerical model. Inputting these traces into the MATLAB prototype allowed us to verify the numerical method directly in the MATLAB environment, as well as providing a dataset for unit-testing the later stages of the development.

5.3.2 C Port

After the numerical method was prototyped and verified in MATLAB, a straightforward C port was performed by one of the computer scientists. Since we anticipated further tuning of the numerical method, we also wrote unit tests for the C port using CppUnit [3]. These tests allowed us to prototype changes in MATLAB, port the changes to the C version, and automatically check that their outputs matched. In this way, the computer scientist could verify the C port against the known-good MATLAB implementation, without requiring in-depth knowledge of the underlying mathematical processes.

5.3.3 Simulated Application

We then integrated the C port of the numerical method into a full TinyOS application, using services provided by the ISHMP framework wherever possible to coordinate the motes over the wireless network. Rather than immediately deploying the implementation on the target hardware, it was first deployed on the TOSSIM simulation environment [20], which allows the execution of full TinyOS applications on a standard PC. TOSSIM provides drivers to simulate basic mote functionality, including realistic radio communication. We added stub drivers to read accelerometer data from disk and to write the SHM output to the PC console. With some care, we were able to write the SHM application code to be portable between TOSSIM and real mote hardware.

Adding this stage to the development process was a late decision that ended up being critical for development. TOSSIM allowed rapidly prototyping changes to the code on a large simulated network; even reinstalling a new codebase on a moderately-sized network of real Imote2 motes is a frustrating and time-consuming process. Perhaps more importantly, the TOSSIM simulation could be directly debugged using the `gdb` source-level debugger and the Valgrind memory debugging toolsuite [28]. These tools allowed us to locate and fix subtle bugs in the network coordination logic that would have been impractical to isolate on the target hardware.

After the application was completed and debugged from end-to-end, the traces used in the previous two stages were fed into the nodes' simulated sensor drivers. This allowed us to verify that the full application's output was identical to the MATLAB prototype.

5.3.4 Deployment on Real Motes

Since the full application had been developed and debugged using the TOSSIM environment, only small changes were needed for bring-up on real Imote2 hardware. One memory bug was discovered and corrected at this stage, which coincidentally did not manifest under TOSSIM because of subtle differences in the network configuration. We also implemented a workaround for a bug in the Imote2's "`newlib`" C library, which parsed configuration input incorrectly on real mote hardware.

6 Evaluation

To validate our system, we implemented and deployed our multi-level damage localization system on three representative structures. We first describe an experiment carried out by injecting data from a simulated truss structure into a testbed of real Imote2 nodes. Experimental results demonstrate that our system is able to accurately localize damage at the member-level. Moreover, latency and energy consumption data collected during the experiment illustrate the efficiency of our decentralized approach. Preliminary results collected in-situ on a real truss structure also validate the approach.

In addition to these experiments, we have validated our system's performance on a simpler cantilever beam structure. Discussion of this experiment is omitted here due to space restrictions, and may be found in [16].

6.1 Simulated Truss

Our first set of experiments involve simulated sensor data from a 5.6 m steel truss structure [11] at the Smart Structure Technology Laboratory at the University of Illinois at Urbana-Champaign. For this structure, we set the sampling frequency to 560 Hz, the record length to 18,432 data points, and the FFT size to 4096 points. We attempted to carry out the experiment on the truss using an earlier implementation of the damage localization system based on the Crossbow ITS400 sensorboard. However, the ITS400 sensor subsystem contains a serious deadlocking bug which prevented us from collecting sufficient vibration data for our experiments.

Instead, we produced two sets of simulated data traces using a finite element model of the truss in MATLAB, with additional measurement noise added to simulate noisy sensor readings. The first set represents the truss in its intact case, providing a baseline flexibility measurement. The second set was generated with simulated damage to three members of the left side of the truss and four members to the right side of the truss. We then

augmented our TinyOS implementation to replay these simulated sensor traces from flash memory, allowing us to inject the traces into live experiments. Timestamping data collected at each stage of the experiment also provided real-world latency data which could be used to derive energy consumption.

Using a network of nine real Imote2 motes and simulated sensor data, our system correctly localized damage to each of the truss's seven affected elements. Because we have since carried out experiments on a real, larger truss (Section 6.2), we focus our discussion here on energy consumption. Further details on the damage localization performance may be found in Appendix B.

6.1.1 Energy Consumption

The timestamping data collected during the experiments provided a direct measurement of the latency in each major stage. We also performed a separate set of experiments to measure the latency of time-synchronizing the motes and collecting 18,432 data samples (since the experiments used replayed data traces). Tables 1 and 2 present the average latencies for the cluster member and cluster head nodes, respectively. Offline, we measured the power draw of each stage using an oscilloscope, which we used to estimate the total energy consumption of each stage in the experiment.

State	Latency (s)	Energy (J)
Synchronization	30.00	12.06
Sensing	53.80	22.96
Compute FDD	21.47	9.28
Transmit FDD	0.21	0.08

TABLE 1
Mean latency and energy cost at cluster member

State	Latency (s)	Energy (J)
Synchronization	40.35	16.23
Sensing	49.68	21.20
Compute own FDD	18.19	7.86
Receive other FDD	0.56	0.23
Compute mode shapes	1.52	0.66
Transmit mode shapes	1.35	0.53

TABLE 2
Mean latency and energy cost at cluster head

Several important observations can be made from this data. First, our decentralized architecture is indeed effective at dramatically reducing the amount of bandwidth and energy consumed in exchanging data among nodes. Our decentralized architecture spends an average of 0.21 s per cluster member exchanging FDD results, plus an average of 1.35 s per cluster head transmitting the mode shape results to the base station. In contrast, based on our prior work [17], we estimate that it would have taken 87 s per sensor to reliably transmit the 18,432 raw sensor readings to the base station for centralized processing.

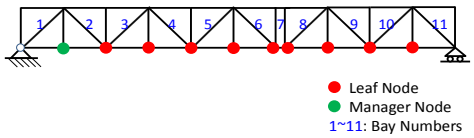
Second, our efficient architecture incurs relatively little overhead on the Imote2 hardware. On the cluster member nodes, as much as 79.4% of the latency and 78.9% of the energy consumption can be attributed to synchronizing the nodes and collecting data. Only 21.1% of the energy consumption represents reducible overhead. The cluster head nodes incur similarly low overheads, with only 20.4% of the latency and 19.1% of the energy consumption attributable to processing and data transmission.

Third, this low overhead leads to low total energy consumption in absolute terms. On average, the cluster member and cluster head nodes consume a total of 44.4 J and 46.7 J, respectively. A power supply of 3x 1.5V, 1250 mAh AAA batteries delivers a theoretical energy supply of 20,250 J. Thus, with proper duty cycling, we anticipate that each node could perform damage localization hundreds of times before depleting its energy supply.

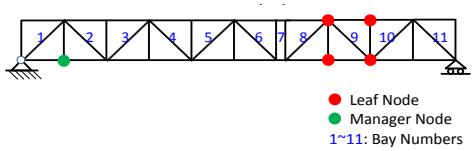
6.2 Real Truss

Since the simulated truss experiments, we have reimplemented the system using a newer version of the ISHMP library and the SHM-A sensorboard (which, crucially, does not suffer from the deadlocks experienced with the ITS400 sensorboard). Using this new implementation, we have carried out preliminary experiments of two-level damage localization on a full-scale steel highway sign support structure at the Bowen Laboratory for Large-Scale Civil Engineering Research at Purdue University. The highway sign truss's properties are described in detail in Appendix C of the supplemental materials.

For the experiment, we installed 19 Imote2 motes on the structure. However, as we discuss below, only a subset of these motes on the back panel of the truss were directly used for damage detection. The remaining motes were used to collect extra data for future studies. The SHM-A sensorboard on each mote was configured to collect data as a sampling frequency of 280 Hz with an associated filter cut-off frequency at 70 Hz. This frequency was best suited to capture the dominant dynamics of this structure.

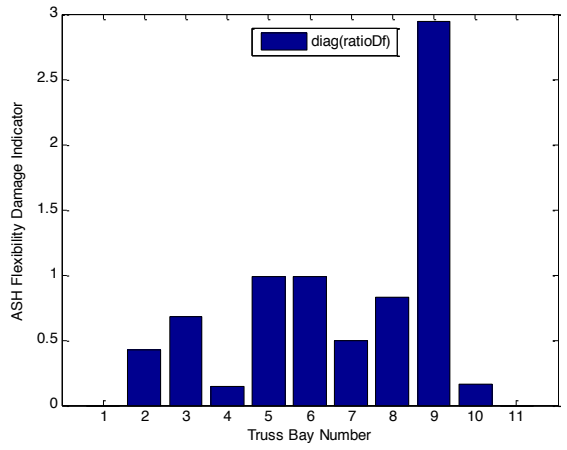


(a) Level 1

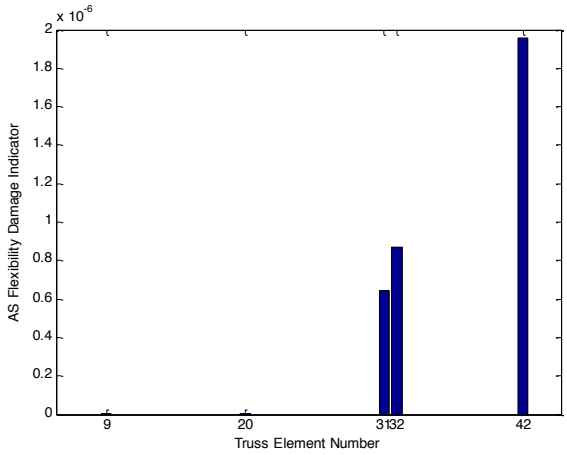


(b) Level 2

Fig. 3. Sensors activated at each level



(a) Level 1



(b) Level 2

Fig. 4. Damage localization results on the real truss

After collecting data from the intact truss, the truss was damaged by cutting halfway through its 42nd element, a diagonal beam on the back panel of the truss's ninth bay. For the first level of damage localization, ten motes were activated along the bottom side of the truss, as shown in Figure 3(a). Each sensor collected 65,536 data points at a frequency of 280 Hz; aggregated data were collected at the base station, where ASHFM was employed to localize damage to the resolution of a bay. These level 1 results, plotted in Figure 4(a), correctly identify damage at the ninth bay.

Based on these results, four sensors surrounding the area of interest were activated, along with a fifth node acting as a manager node (shown in Figure 3(b)). New data were collected at the same sampling rate and size for a second level of damage localization, and ASFM was used at the base station to localize damage to a specific element. These level 2 results, plotted in Figure 4(b), correctly localize damage to element 42. Both levels of damage localization match simulation results and the actual damage scenario, validating both the numeric approach and its distributed implementation.

7 Conclusion

Structural health monitoring of civil infrastructure represents an important application domain of cyber-physical systems. We propose a novel cyber-physical co-design approach to structural health monitoring based on wireless sensor networks. Our distributed structural health monitoring system integrates (1) flexibility-based structural engineering methods that can localize damages at different resolution and costs, and (2) an efficient, *multi-level* computing architecture that leverage on the multi-resolution feature of flexibility-based methods. A key feature of our approach is that it selectively activates nodes in the damaged region in order to achieve fine-grained localization damage localization while allowing many of the nodes to remain asleep. We have implemented our approach on the Intel Imote2 hardware platform and the TinyOS operating system. Experimental results show that our system is able to localize damage to the resolution of a single element on a representative simulated and real truss structures. We also demonstrate the energy efficiency of this approach through latency and energy consumption measurements. Our results illustrate the promise of cyber-physical approach which consider both the architecture of the cyber (wireless sensor network) system and the characteristics of the physical (structural engineering) methods.

Acknowledgments

This work is supported in part by NSF through grants CNS-0627126, CNS-1035773, CNS-1035748 and CNS-0708460. We would like to thank Prof. B.F. Spencer and Shin Ae Jang for their help with our experiments related to the simulated truss, and the ISHMP team at UIUC for making the software library available.

References

[1] http://www.tinyos.net/.
[2] http://shm.cs.uiuc.edu/.
[3] http://cppunit.sourceforge.net/doc/lastest/.
[4] A. A. Abbasi and M. Younis. A survey on clustering algorithms for wireless sensor networks. *Comput. Commun.*, 30(14-15):2826–2841, Oct. 2007.
[5] American Society of Civil Engineering. 2009 report card for America's infrastructure. http://www.asce.org/reportcard/2009/.
[6] D. Bernal and B. Gunes. Flexibility based approach for damage characterization: Benchmark application. *Journal of Engineering Mechanics*, 130(1):61–70, January 2004.
[7] N. Castaneda, F. Sun, S. Dyke, C. Lu, A. Hope, and T. Nagayama. Implementation of a correlation-based decentralized damage detection method using wireless sensors. In *ASEM Conference*, 2008.
[8] N. Castaneda, G. Yan, and S. Dyke. Evaluation of the performance of a distributed structural health monitoring algorithm for wireless sensing, 2009.
[9] M. Ceriotti, L. Mottola, G. P. Picco, A. L. Murphy, S. Guna, M. Corra, M. Pozzi, D. Zonta, and P. Zanon. Monitoring heritage buildings with wireless sensor networks: The torre aquila deployment. In *IPSN*, 2009.
[10] K. Chebrolu, B. Raman, N. Mishra, P. K. Valiveti, and R. Kumar. BriMon: a sensor network system for railway bridge monitoring. In *MobiSys*, 2008.

[11] E. Clayton. Development of an experimental model for the study of infrastructure preservation. In *National Conference on Undergraduate Research*, 2002.

[12] Crossbow Technology. MICAz wireless measurement system. http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICAz_Datasheet.pdf.

[13] Crossbow Technology, Inc. *Imote2 Hardware Reference Manual*, 2007.

[14] Z. Duan, G. Yan, and J. Ou. Structural damage detection using the angle-between-string-and-horizon flexibility (under review). *Structural Engineering and Mechanics*.

[15] M. Gangone, M. Whelan, K. Janoyan, K. Cross, and R. Jha. Performance monitoring of a bridge superstructure using a dense wireless sensor network. In *International Workshop on Structural Health Monitoring, Stanford, California*, 2007.

[16] G. Hackmann, W. Guo, G. Yan, C. Lu, and S. Dyke. Cyber-physical codesign of distributed structural health monitoring with wireless sensor networks. In *ICCPS*, 2010.

[17] G. Hackmann, F. Sun, N. Castaneda, C. Lu, and S. Dyke. A holistic approach to decentralized structural damage localization using wireless sensor networks. In *RTSS*, 2008.

[18] A. Jindal and M. Liu. Networked computing in wireless sensor networks for structural health monitoring. *IEEE/ACM Trans. Netw.*, 20(4):1203–1216, Aug. 2012.

[19] S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, and M. Turon. Health monitoring of civil infrastructures using wireless sensor networks. In *IPSN*, 2007.

[20] P. Levis, N. Lee, M. Welsh, and D. Culler. TOSSIM: accurate and scalable simulation of entire TinyOS applications. In *SenSys*, 2003.

[21] B. Li, D. Wang, F. Wang, and Y. Q. Ni. High quality sensor placement for SHM systems: Refocusing on application demands. In *INFOCOM*, 2010.

[22] X. Liu, J. Cao, S. Lai, C. Yang, H. Wu, and Y. Xu. Energy efficient clustering for wsn-based structural health monitoring. In *INFOCOM*, 2011.

[23] J. Lynch and K. Loh. A summary review of wireless sensors and sensor networks for structural health monitoring. *Shock and Vibration Digest*, 38(2):91–128, 2006.

[24] M. Maróti, B. Kusy, G. Simon, and A. Lédeczi. The flooding time synchronization protocol. In *SenSys*, 2004.

[25] A. Messina, I. A. Jones, and E. J. Williams. Damage detection and localization using natural frequency changes. In *Conference on Identification in Engineering Systems, U.K.*, 1996.

[26] D. Moss and P. Levis. BoX-MACs: Exploiting physical and link layer boundaries in low-power networking. Technical Report SING-08-00, Stanford Information Networks Group, 2008.

[27] T. Nagayama. *Structural Health Monitoring Using Smart Sensors*. PhD thesis, University of Illinois at Urbana-Champaign, 2007.

[28] N. Nethercote and J. Seward. Valgrind: A framework for heavy-weight dynamic binary instrumentation. In *PLDI*, 2007.

[29] S. N. Pakzad, G. L. Fenves, S. Kim, and D. E. Culler. Design and implementation of scalable wireless sensor network for structural monitoring. *ASCE Journal of Infrastructure Engineering*, 14(1):89–101, March 2008.

[30] A. Pandey and M. Biswas. Experimental verification of flexibility difference method for locating damage in structures. *Journal of Sound and Vibration*, (184), 1995.

[31] J. Polastre, R. Szewczyk, and D. Culler. Telos: Enabling ultra-low power wireless research. In *IPSN*, 2005.

[32] J. A. Rice and B. F. Spencer, Jr. Structural health monitoring sensor development for the Imote2 platform. In *Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems*, 2008.

[33] W. Song, S. Dyke, G. Yun, and T. Harmon. Improved damage localization and quantification using subset selection. *Journal of Engineering Mechanics*, 135(6):548–560, 2009.

[34] B. Spencer and T. Nagayama. Smart sensor technology: a new paradigm for structural health monitoring. In *Asia-Pacific Workshop on Structural health Monitoring, Yokohama, Japan*, 2006.

[35] F. Wang, D. Wang, and J. Liu. Elesense: Elevator-assisted wireless sensor data collection for high-rise structure monitoring. In *INFOCOM*, 2012.

[36] J.-H. Weng, C.-H. Loh, J. P. Lynch, K.-C. Lu, P.-Y. Lin, and Y. Wang. Output-only modal identification of a cable-stayed bridge using wireless monitoring systems. *Engineering Structures*, 30(7):1820–1830, 2008.

[37] N. Xu, S. Rangwala, K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin. A wireless sensor network for structural monitoring. In *SenSys*, 2004.

[38] G. Yan, Z. Duan, and J. Ou. An axial strain flexibility for damage detection of truss structure. In *International Workshop on Smart Materials and Structures*, 2005.

[39] G. Yan, S. J. Dyke, and A. Irfanoglu. Experimental validation of a damage detection approach on a full-scale highway sign support truss. *Mechanical Systems and Signal Processing*, 2011.

[40] G. J. Yun, K. A. Ogorzalek, S. J. Dyke, and W. Song. A two-stage damage detection approach based on subset selection and genetic algorithms. *Smart Structures and Systems*, 5:1–21, 2009.

[41] A. T. Zimmerman, M. Shiraishi, R. A. Swartz, and J. P. Lynch. Automated modal parameter estimation by parallel processing within wireless monitoring systems. *Journal of Infrastructure Systems*, 14(1):102–113, March 2008.

Gregory Hackmann received a Ph.D. in Computer Science from Washington University in St. Louis in 2011. His research interests include wireless sensor networks and embedded systems. He is currently a software engineer at Google.

Weijun Guo received an M.S. degree in Computer Science from Washington University in St. Louis in 2009.

Guirong Yan is an assistant professor at University of Texas at El Paso. Her research covers structural health monitoring (SHM) and damage detection, sensor technologies and smart materials. Her research subjects include long-span bridges, high-rise buildings, offshore platforms and subsea pipelines. She has been involved in field tests of large-scale civil infrastructure systems and practical applications of SHM techniques.

Zhuoxiong Sun received his B.S. and M.S. degrees in Mechanical Engineering from Zhejiang University, China, and Purdue University, USA, in 2010 and 2012, respectively. He is currently pursuing his Ph.D. in Mechanical Engineering from Purdue University. His current research interests include wireless structural control and wireless structural health monitoring. He is a student member of the ASCE and the ASME.

Chenyang Lu is a Professor of Computer Science and Engineering at Washington University in St. Louis. He received the Ph.D. degree from University of Virginia in 2001, the M.S. degree from Chinese Academy of Sciences in 1997, and the B.S. degree from University of Science and Technology of China in 1995, all in computer science. His research interests include real-time systems, wireless sensor networks and cyber-physical systems.

Shirley Dyke is Professor of Mechanical Engineering and Civil Engineering, School of Mechanical Engineering, Purdue University and the director of the Intelligent Infrastructure Systems Lab. Dr. Dyke focuses on the use of cyberinfrastructure and advanced technologies such as structural control and monitoring systems for improving the lifecycle performance of structural systems.