Computer-Aided Civil and Infrastructure Engineering **25** (2010) 504–516

Mobile Agent Computing Paradigm for Building a Flexible Structural Health Monitoring Sensor Network

Bo Chen*

Department of Mechanical Engineering—Engineering Mechanics, Department of Electrical and Computer Engineering, Michigan Technological University, Houghton, MI, USA

&

Wenjia Liu

Department of Electrical and Computer Engineering, Michigan Technological University, Houghton, MI, USA

Abstract: *Wireless structural health monitoring research has drawn great attention in recent years from various research groups. While sensor network approach is a feasible solution for structural health monitoring, the design of wireless sensor networks presents a number of challenges, such as adaptability and the limited communication bandwidth. To address these challenges, we explore the mobile agent approach to enhance the flexibility and reduce raw data transmission in wireless structural health monitoring sensor networks. An integrated wireless sensor network consisting of a mobile agent-based network middleware and distributed high computational power sensor nodes is developed. These embedded computer-based high computational power sensor nodes include Linux operating system, integrate with open source numerical libraries, and connect to multimodality sensors to support both active and passive sensing. The mobile agent middleware is built on a mobile agent system called Mobile-C. The mobile agent middleware allows a sensor network moving computational programs to the data source. With mobile agent middleware, a sensor network is able to adopt newly developed diagnosis algorithms and make adjustment in response to operational or task changes. The presented mobile agent approach has been validated for structural damage diagnosis using a scaled steel bridge.*

1 INTRODUCTION

Structural health monitoring (SHM) is an emerging technology in civil, mechanical, and aerospace engineering to detect damage in structures (He et al., 2008; Li and Wu, 2008; Moaveni et al., 2008; Psimoulis and Stiros, 2008; Sohn et al., 2008). The SHM process typically involves the observation of the dynamic response of a structure from a group of sensors, the extraction of damage-sensitive features from these measurements, and analysis of these features to determine the current state of the structure (Kolakowski, 2007). Because the structural damage is an intrinsically local phenomenon, responses from sensors close to the damaged location are expected to be more heavily affected than those far away from the damage site (Nagayama et al., 2009). For complicated structures, a sensor network, with onboard computation and wireless communication capabilities, densely deployed over the entire structure has the potential to provide rich information for effective damage diagnosis and localization.

Although sensor network approach is suitable for SHM, the design of wireless sensor networks presents

---

*To whom correspondence should be addressed. E-mail: *bochen@mtu.edu*.

DOI: 10.1111/j.1467-8667.2010.00656.x

a number of challenges. (1) Adaptability: Sensor networks suffer substantial network dynamics due to node failure, added new nodes, environmental obstructions, and user demand changes. A sensor network should be able to make appropriate adjustments to operate robustly when the environment and network itself change (Römer, 2004). (2) Distributed data processing and damage diagnosis: Due to the high sampling frequency, an SHM sensor network generates a huge amount of measurement data during the monitoring process. If all the sensor data are centrally processed, these data need to be sent to a central station. Transmitting this large amount of data over a wireless sensor network is challenging because of the significant limitation of communication bandwidth. To reduce the raw data transmission and the response time, a number of researchers have proposed distributed data processing in SHM sensor networks (Gao et al., 2006). (3) Scalability: Scalability is the ability of a sensor network to allow the growth of the number of sensor nodes without affecting the performance of the network (Hadim and Mohamed, 2006a). Scalability is a desirable property of a sensor network as the size of the required network is usually unknown at the design stage. The sensor network should maintain at an acceptable performance level as the network grows for a larger sensing area or higher resolution. (4) Self-organization: For large structures, sensor networks usually consist of thousands of nodes and may be deployed in unreachable environments (embedded in physical structure). Having such a deployment size and environment, it is impossible to pay special attention to any individual node. Self-organization is a key issue in the design of sensor networks (Blumenthal et al., 2003). (5) Multitasking: Most existing sensor networks were designed to be application specific. However, it is widely accepted that sensor networks will have long deployment cycles serving multiple transient users with dynamic needs (Boulis et al., 2003). In addition, multiple applications (tasks) may be performed concurrently over a single-sensor network. For example, a building monitoring system may need to simultaneously monitor the temperature and luminance, check cracks on the wall, track traversing persons, and even communicate with systems in nearby buildings (Yu et al., 2004).

A large number of papers have been published on the applications of agents in recent years mostly outside civil engineering (Chen et al., 2008a; Monticolo et al., 2008; López-París and Brazález-Guerra, 2009). To address the aforementioned challenges, this article presents a mobile agent-based framework that pursues desirable characteristics, such as adaptability, distributed damage diagnosis, and sensor node collaboration. The major design considerations of the presented sensor network framework are as follows. (1) Sensor node design: possess high computational power; equip with multimodality sensors; open source Linux operating system (OS); and open source software implementation; (2) Network middleware design: reduce network traffic by moving computational algorithms instead of sensor data; support generation and migration of mobile monitoring agents; allow collaboration in local sensor communities and collaborative distributed data processing; self-organize through mutual interaction among agents to agents and agents to the environment.

The rest of the article is organized as follows. Section 2 reviews the state of the art of structural health monitoring systems and damage diagnosis methodologies. Section 3 presents the hardware and software design of sensor nodes. Section 4 introduces a mobile agent system and the use of this system as a sensor network middleware. Section 5 illustrates the deployment of damage diagnosis algorithms on sensor nodes via mobile agents. Section 6 discusses several practical issues of using a mobile agent approach. Finally, conclusions are made in Section 7.

2 RELATED WORK

This section describes the background of sensor network system design and damage detection methodologies. Lynch and Loh (2006) gave a summary review of wireless sensors and sensor networks for structural health monitoring. Research in this area, including hardware design of wireless sensor nodes, embedded software for wireless sensors, and emerging wireless sensor concepts, were introduced. Spencer et al. (2004) provided the state-of-the-art review of current “smart sensing” technologies in the SHM area. Farrar et al. (2006a) summarized and compared several sensor network systems for the structural health monitoring. Tanner et al. (2003) developed a proof of concept SHM system using off-the-shelf hardware, “Motes” running on TinyOS operating system. Due to limited resources available in the processor board, only the most rudimentary data interrogation algorithms were implemented in the system. Lynch et al. (2002) presented a hardware sensor unit for a wireless peer-to-peer SHM system. Using off-the-shelf components, the authors combine sensing circuits and wireless transmission with a computational core for the decentralized data collection, analysis, and broadcast monitoring results. The embedded software is tightly integrated with the hardware. Nagayama et al. (2007) used a new generation of Mote, Imote2, as a hardware platform and implemented several SHM algorithms in their sensor units to promote distributed computing strategy (Gao et al., 2006). To increase the node processing power, Farrar et al. (2006b) selected a single-board computer integrated with a digital signal processing board and a wireless network board to construct a prototype sensing system. They also integrated Matlab-based data interrogation functions into this single-board computer-based sensor hardware.

The vibration-based damage assessment of bridge structures and buildings has been studied since the early 1980s. A number of research results have been reported in the literature (Carden and Brownjohn, 2008; Soyoz and Feng, 2009). Doebling et al. (1996) reviewed research on vibration-based damage identification and health monitoring. Sohn et al. (2003) reviewed technical papers in structural health monitoring, published between 1996 and 2001. Most conventional structural health monitoring methods are modal analysis based. Modal parameters, such as natural frequencies, damping ratios, and mode shape curvature, have been the primary features used to identify damage in structures. Recently, a number of new approaches, such as wavelet-based (Pakrashi et al., 2007; Su et al., 2007), neural network-based (Jiang and Adeli, 2008a, 2008b) and pattern recognition-based (Sohn and Farrar, 2001; Chen and Zang, 2009), have been developed for health monitoring of structures. For example, Adeli and Jiang (2006) presented a new dynamic time-delay fuzzy wavelet neural network model for nonparametric identification of structures. The integration of four computing concepts: dynamic time delay neural network, wavelet, fuzzy logic, and the reconstructed state space concept from the chaos theory, provided a quick training convergence and improved system identification accuracy. To further enhance training convergence and numerical accuracy, the authors developed an adaptive Levenberg–Marquardt least-squares algorithm with a backtracking inexact linear search scheme (Jiang and Adeli, 2005) to speed up training process and proposed a Bayesian discrete wavelet packet transform denoising approach (Jiang et al., 2007) for accurate structural system identification. Jiang and Adeli (2007) presented a new damage evaluation method based on a power density spectrum method, called *pseudospectrum*. They developed a MUSIC (multiple signal classification) method for computation of the pseudospectrum from the structural response time series and applied it to data obtained for a 38-storey concrete test model. Sohn and Farrar (2001) proposed a statistical pattern recognition method for damage diagnosis using time-series analysis of vibration signals. The residual error ratio of autoregressive (AR) with exogenous input (ARX) models for test signal and the reference signal is defined as the damage-sensitive feature. Park et al. (2007) presented a novel approach for health monitoring of structures using terrestrial laser scanning. Chen and Zang (2009) presented an Artificial Immune Pattern Recognition approach for the damage classification in structures. The structural damage pattern recognition is achieved through mimicking immune recognition mechanisms that possess features such as adaptation, evolution, and immune learning. The damage patterns are represented by feature vectors that are extracted from the dynamic response of a structure.

3 SENSOR NODE HARDWARE AND SOFTWARE DESIGN

Sensor nodes are building blocks of wireless sensor networks. For the SHM sensor networks, the desirable characteristics of sensor nodes are as follows. First, high computational power sensor nodes are highly recommended. Local data processing can reduce the raw data transmission over a network. The reduction of data transmission can save network bandwidth and energy. The energy cost of sending one single bit of data can consume the energy executing thousands of instructions to produce the same data (Hadim and Mohamed, 2006b). Second, open software implementation is desirable to promote software reuse. The open software architecture allows user communities to participate in improving node functionalities and developing new software. Third, multimodality sensors help to achieve a better assessment of the structural state from a comprehensive view of the structure. Finally, reprogrammable sensors are welcome to increase the adaptability and support the multitasking purposes.

Having the aforementioned node design criteria in mind, we chose a finger size embedded computer called Gumstix (Gumstix, 2009) as sensor node computing platform. The sensor node consists of three boards as shown in Figure 1. The sensing board lies at the bottom; the Gumstix board is located in the middle; and a wireless communication board sits at the top. Three boards are connected together through predesigned connectors. The Gumstix board communicates with the sensing board through I^2C bus, and connects to the wireless communication board through a parallel port. The volume of the sensor node is about $4 \times 2.4 \times 0.65$ in^3.

The high computational power of the sensor node is achieved through the integration of sensor node hardware computing resources and the embedded numerical computing software packages. The Gumstix embedded computer is one of the world's smallest full function miniature computers with a size of 20 mm × 80 mm × 8 mm. The product is based on the Intel PXA-255 processor with Xscale technology and a Linux operating

Fig. 1. A high computational power sensor node.

system. The low cost and high performance make it a good candidate for the embedded applications. The Gumstix maximum on-board memory sizes are 128 MB RAM, 32 MB flash, and the CPU speed can reach 600 MHz. The Gumstix board that we used has 64 MB RAM, 16 MB Flash, and 400 MHz CPU speed. Gumstix family expansion boards also provide external memory spaces. For example, the WiFi card contains a Type II compact Flash adapter, providing an ample storage space for embedding software algorithms. This memory space is directly accessible through Gumstix file system. Gumstix embedded computer is governed by a multitask general-purpose Linux OS stored in the onboard flash memory. Two server programs, a remote secure shell server and a web server, are provided for the users to remotely access the computer. The application software is compiled using GNU Compiler Collection (GCC) cross-compiler and downloaded to the Gumstix for execution. The Gumstix computers have earned a wide range of applications, such as radio-frequency identification, sensor management, control panels, personnel management devices, reading tablets, network security, software appliances, robotics, unmanned aerial vehicle, and many more areas of engineering and business. Gumstix computers can connect to a network in many ways through its extension boards: over Universal Serial Bus (USB) or serial port, by using Transmission Control Protocol/Internet Protocol (TCP/IP) over a Bluetooth protocol service, with 10/100 Ethernet, or via WiFi.

A custom sensor board is designed and fabricated by our research group to meet the structural health monitoring sensing requirement. We employ multimodal sensing approach and incorporate active sensing with passive sensing to achieve a better monitoring result. The sensor board consists of an Atmega128L CPU for real-time data acquisition and communication with the Gumstix mother board, 16-bit analog/digital (A/D) converters and signal conditioning circuits for accelerometer and strain gage signal processing, an active sensing signal generator and response analyzer for active sensing with Piezoelectric Transducer (PZT) sensors/actuators, a ZigBee Module for low-power wireless communication, and an external Static Random Access Memory (SRAM) for real-time data buffering.

To facilitate the implementation of damage diagnosis algorithms on sensor nodes, a number of numerical libraries are integrated into sensor nodes. Thanks to the open source software packages Ch (Ch, 2009), CLAPACK (Anderson et al., 1999), and Numerical Recipes in C (Press et al., 1992), which make it easy to perform damage diagnosis on sensor nodes and build an open software architecture. Ch is an embeddable C/C++ interpreter. It supports matrix computation and provides a set of high-level numerical analysis functions for data analysis. Ch is also the execution engine of mobile agents in the presented mobile agent-based network framework. The CLAPACK library is a C version of LAPACK library that provides routines for solving systems of linear equations, linear least-squares problems, eigenvalue problems, and singular value problems (Anderson et al., 1999). All the functions support real and complex matrices, in both single and double precision. Numerical Recipes in C is another good tool for people who program in C and work with mathematics. It covers a wide range of algorithms. Routines are included from solving systems of linear equations to determining eigenvectors and singular value decompositions, solving differential equations, and calculating Fast Fourier Transforms.

The sensor node software consists of two layers as shown in Figure 2. The upper layer data processing and SHM algorithms are implemented in Gumstix, while the sensor data acquisition software is implemented in sensing board microcontroller. The upper layer software adopts open source and modular implementation. The Numerical Libraries and Utility Functions provide computational building blocks to construct SHM analysis algorithms. The utility functions are designed to perform a certain subtask of SHM analysis or common computation that is not available in numerical libraries, for example, Fast Fourier Transform. The existing open source numeric libraries such as Numerical Recipes in C (Press et al., 1992) can be very helpful to the implementation of these utility functions. A Mobile-C-based mobile agent middleware supports the execution and migration of mobile monitoring agents. The serial communication and WiFi communication modules communicate with the sensing board and remote entities through I^2C and WiFi communication protocols.

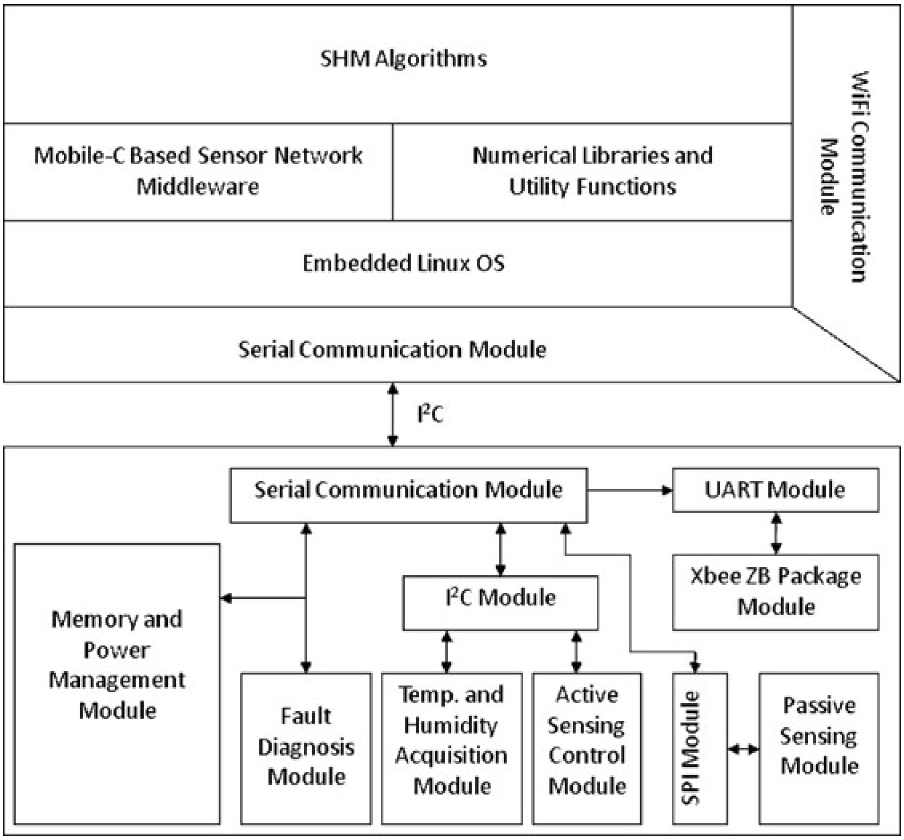

Fig. 2. Two-layer sensor node software design.

The lower layer embedded software manages data acquisition and sensing board communication. The passive data acquisition is handled in the timer interrupter processing module. Active sensing control module uses I^2C serial communication to transmit data and send commands, while passive sensing module communicates with the microcontroller via Serial Peripheral Interface (SPI) bus. Temperature and humidity acquisition module uses Sensibus (a communication protocol similar to I^2C) to communicate with the Microcontroller.

4 A MOBILE AGENT-BASED NETWORK MIDDLEWARE FOR STRUCTURAL HEALTH MONITORING SENSOR NETWORKS

Programming sensor networks is currently a cumbersome and error-prone task as it requires programming individual sensor nodes using low-level programming languages and needs to interface with the sensor hardware and the network (Römer, 2004). In addition, most of the time, it is assumed that the algorithms are hard-coded into the memory of each node. Although some platforms allow the application developers using a node-level OS to create the application, the developer still has to create a single executable image to be downloaded manually into each node (Boulis et al., 2003). There is a strong need for developing middleware that simplifies tasking sensor networks and supports dynamic programming sensor networks.

To overcome the aforementioned problems, a number of middleware approaches are currently being investigated by researchers in the community to provide dynamic programming environments. Some of these approaches are inspired by mobile code (Levis and Culler, 2002; Boulis et al., 2003; Szumel et al., 2005; Chen, 2008). Maté (Levis and Culler, 2002) is a byte code interpreter that runs on TinyOS, an OS specifically designed for sensor networks that run on motes. Application programs are broken up into small capsules of 24 instructions, each of which is a single-byte long. Large programs can be composed of multiple capsules. The capsules can self-replicate through the network. Sending and reception capsules enable the deployment of *ad hoc* routing and data aggregation algorithms. However, Maté's ability to allow code motion is limited (Szumel et al., 2005). It propagates a single program by comparing program versions between neighbors and

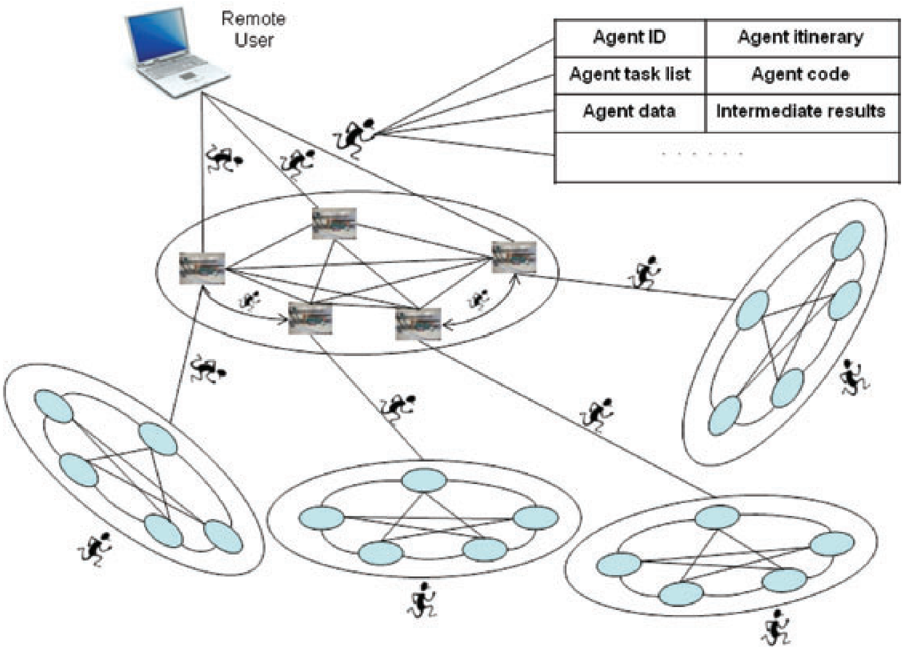

Fig. 3. A mobile agent-based structural health monitoring sensor network.

updating the older program from the newer one. SensorWare (Boulis et al., 2003) is another work pursing dynamic programming of sensor networks. In SensorWare, programs are coded in Tcl scripts. The replication of such scripts in several sensor nodes allows the dynamic deployment of distributed algorithms into the network. While SensorWare supports the implementation of arbitrary queries, even simple sensing tasks result in complex scripts that have to interface with OS functionality and the network (Römer, 2004).

This article presents a mobile agent approach for building a sensor network platform to reduce data transmission and enhance the flexibility of distributed structural health monitoring systems. Taking advantage of the mobility of a mobile agent system, the presented agent platform allows moving diagnosis programs to data sources and performing damage diagnosis locally as shown in Figure 3. The distributed sensor nodes can dynamically accept mobile agents for the deployment of new damage diagnosis algorithms and sensing strategies in response to the changes of monitoring conditions. In a mobile agent-based SHM sensor network, a remote user can dispatch mobile agents to sensor nodes in the network. Mobile agents carrying code and execution states move from one sensor node to another, read sensor data, perform damage diagnosis on the sensor nodes where they reside, and send diagnosis results back to the remote users. Each agent has its own identification number that is assigned to the agent when it is created. This number will accompany the agent for the entire life of the agent. Agent migration is achieved through message passing. When a mobile agent is dispatched, information related to the agent such as agent ID, agent itinerary, tasks to be performed, and agent code for each task, is encapsulated into a mobile agent message. The intermediate results from each task will be added into the mobile agent message when the agent travels. Finally, the mobile agent will send all the results back to the dispatcher.

To support mobile agent generation, migration, execution, and management, the presented mobile agent-based sensor network platform is developed based on a mobile agent system called Mobile-C (Chen et al., 2006; Chen et al., 2008b; Chen et al., 2009). Mobile-C is an IEEE FIPA (FIPA, 2009) compliant mobile agent system supporting mobile C/C++ agents. It has a small footprint and is easy to be integrated with resource-constrained systems, such as sensor networks. In the presented mobile agent-based sensor network, each sensor node has Mobile-C installed on the Gumstix board as shown in Figure 4. Commonly used numerical functions for SHM algorithms are also integrated into sensor nodes to achieve a small size of mobile agent code for data processing and damage diagnosis. The sensing and signal conditioning board connects to different types of sensors to acquire real-time structural parameters, such as acceleration, strain, stress, temperature, and humidity. A wireless communication board is designed for the communication among distributed sensor nodes. The Mobile-C in sensor nodes can host both

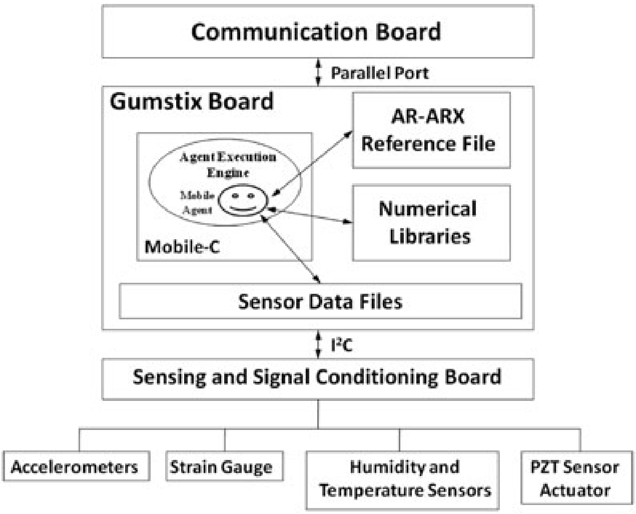

Fig. 4. An SHM sensor node integrated with a mobile agent middleware.

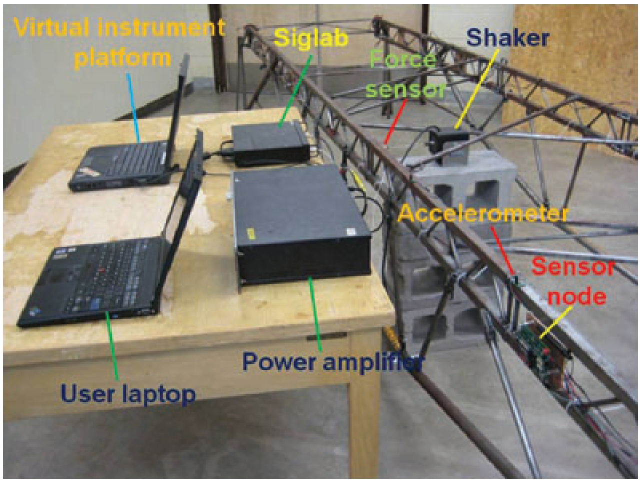

Fig. 5. Test bridge structure.

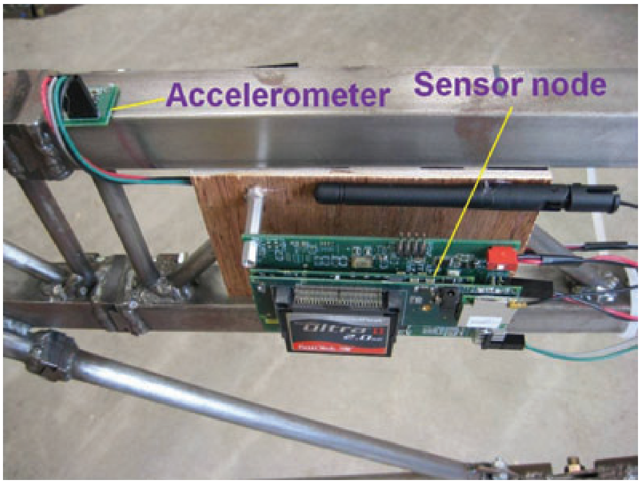

Fig. 6. Sensor node and accelerometer.

stationary agents and mobile agents. Stationary agents are those staying in the sensor nodes where they are created, such as data acquisition agents and regional or central management agents. Mobile agents are those created during the system operation and able to move to different sensor nodes in a network. Different types of mobile agents could be created and dispatched to sensor nodes as needed. For example, the central station could dispatch mobile alert agents to sensor nodes for monitoring specified events. Data analysis and damage diagnosis mobile agents with certain expertise (equipped with different data analysis and damage diagnosis algorithms) can roam over the network to perform monitoring tasks.

5 DYNAMIC DEPLOYMENT OF DAMAGE DIAGNOSIS ALGORITHMS ON SENSOR NODES VIA MOBILE AGENTS

To demonstrate the ability of dynamically deploying SHM algorithms on sensor nodes via mobile agents, this section gives an example of sending two mobile agents to remote sensor nodes to perform damage diagnosis based on local sensor data.

5.1 Experimental setup

A scaled steel bridge shown in Figure 5 was used for the mobile agent validation test. The bridge has two side beams and eight cross-members. Each side beam is composed of six beam sections. Cross-members are distributed near the connections of side beams with two members crossed at the center of the bridge. Accelerometers were mounted on the top of side beams as shown in Figure 6. The outputs of accelerometers were connected to A/D converters on the sensor board nearby.

During the test, the bridge was excited by a shaker at the center of the bridge as shown in Figure 5. Figure 7 shows the excitation and force sensing loop. Siglab and virtual instruments were chosen to generate and monitor the excitation signals of the shaker. Siglab system is seamlessly integrated with MATLAB. Virtual instruments running in the MATLAB environment include classes of Network Analyzer, Function Generator, Spectrum Analyzer, and Oscilloscope. For the bridge test, we used the Function Generator to generate excitation signals for the shaker and Network Analyzer to measure the signals from the force sensor. The shaker excitation signals generated by the Function Generator

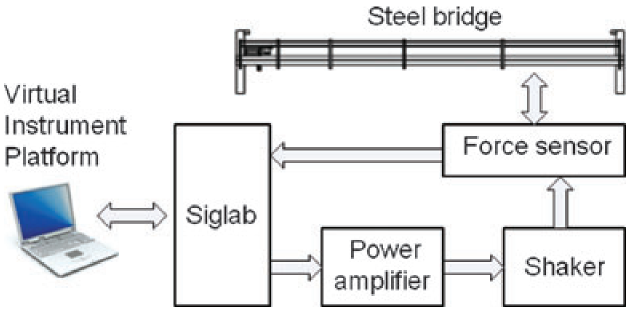

Fig. 7. Shaker and excitation signal generation.

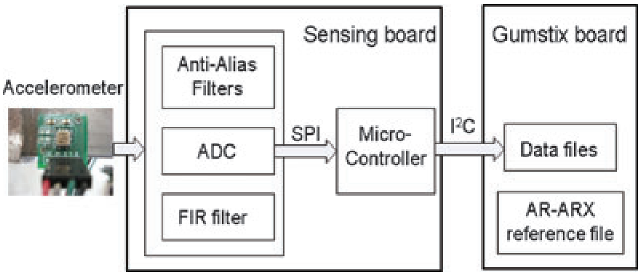

Fig. 8. Acceleration signal conditioning and data transmission between the sensing board and the Gumstix board.

were amplified by a power amplifier. Both the shaker and power amplifier are made by the labworks company. A force sensor was attached to the shaker. The output of the force sensor was fed back to the Siglab and displayed in the Graphical User Interface of the virtual instruments on the laptop.

Figure 8 shows the acceleration data collection, signal conditioning, and data transmission between the sensing board and the Gumstix board. Acceleration data were sampled at a rate of 125 sps. To avoid sample-rate fluctuation and signal aliasing, a programmable signal conditioner Quickfilter, QF4A512 (Quickfilter, 2008), was used for signal conditioning and A/D conversion of accelerometer measurements. This programmable signal conditioner has 4-channel 12/16-bit resolution A/D converters, programmable gain of the amplifier, analog antialiasing filter with 500 kHz cutoff frequency, individually selectable sampling frequencies and individually programmable digital FIR filter. Rice and Spencer (2008) validated the performance of QF4A512 in the field of structural health monitoring. Microcontrollers on the sensing boards read acceleration data from A/D converters through an SPI interface. The collected acceleration data were transmitted to the Gumstix board and saved into data files on the Gumstix board. The interboard communication between the Gumstix board and the sensing board is achieved by I^2C serial communication.

5.2 AR and ARX damage diagnosis algorithm

The damage diagnostic method selected is AR and ARX models proposed by Sohn and Farrar (2001). This two-stage prediction method firstly uses an AR model as shown in Equation (1) to fit a discrete time series of acceleration data $x(k)$. The structural response data at time $t = k\Delta t$, $x(k)$, is a function of p previous response data plus the error term $e_x(k)$. Weights on previous response data are AR coefficients. Because the error $e_x(k)$ in Equation (1) is also affected by unknown external inputs, an ARX model is used in the second stage to establish the relationship between time signal $x(k)$ and AR model error $e_x(k)$, as shown in Equation (2). The term $\varepsilon_x(k)$ is the residual error of the ARX model.

$$x(k) = \sum_{i=1}^{p} c_{xi} x(k-i) + e_x(k) \tag{1}$$

$$x(k) = \sum_{i=1}^{a} \alpha_i x(k-i) + \sum_{j=0}^{b} \beta_j e_x(k-j) + \varepsilon_x(k) \tag{2}$$

To use AR-ARX method for damage diagnosis, a reference file that contains AR and ARX prediction model pairs is required. These AR and ARX predication models are constructed based on discrete time data sets representing the undamaged structure. During damage diagnosis, AR coefficients are computed with Equation (3) using measured discrete time acceleration data $y(k)$ from the monitoring structure. Next, the identification of an ARX model in the reference file is conducted by matching the measured AR model with an AR model in the reference file based on the minimum distance measure shown in Equation (4). The counterpart (ARX model) of the matched AR model in the reference file is used to calculate the residual errors of measured data set using Equation (5). The ratio $\frac{\sigma(\varepsilon_y)}{\sigma(\varepsilon_x)}$ is defined as a damage sensitive feature, where σ is the standard deviation of the residual time series. An appropriate threshold of this ratio is chosen to minimize false-positive and false-negative damage identification.

$$y(k) = \sum_{i=1}^{p} c_{yi} y(k-i) + e_y(k) \tag{3}$$

$$Distance = \sum_{i=1}^{p} (c_{xi} - c_{yi})^2 \tag{4}$$

$$\varepsilon_y(k) = y(k) - \sum_{i=1}^{a} \alpha_i y(k-i) - \sum_{j=0}^{b} \beta_j e_y(k-j) \tag{5}$$

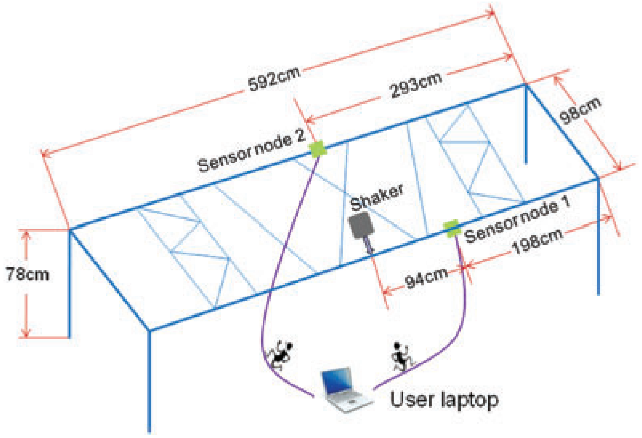

Fig. 9. Sensor node locations (not to scale).

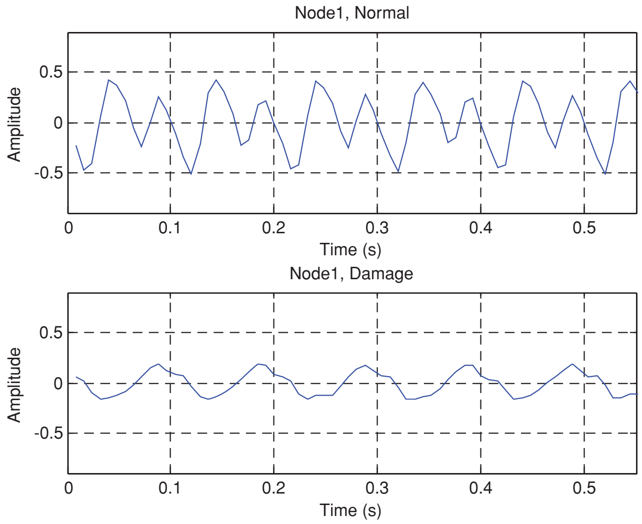

Fig. 10. Sensor node 1 acceleration signals.

5.3 Sensor node locations and AR-ARX reference file

To validate the dynamic deployment of mobile agents for damage diagnosis, sensor nodes 1 and 2 on the scaled steel bridge were chosen for the validation test. Figure 9 shows the size of the steel bridge and locations of sensor nodes. Structural damage was simulated by removing two cross-members at the center of the bridge. For both of normal and damage patterns, acceleration data collected by sensor nodes 1 and 2 were recorded. The excitation signals applied to the shaker were sine waves with peak-to-peak voltage of 0.255, 0.275, and 0.295 V, respectively. At each excitation voltage level, a total of 26 acceleration time series on each sensor node were recorded for the normal pattern, while 19 time series were recorded for the damage pattern. The AR-ARX reference file was constructed by using 10 normal acceleration time series. The standard deviation of the residual time series of rest 16 time series for the normal pattern acceleration data and 19 time serials for the damage pattern were calculated off-line. The values of the ratio, $r = \frac{\sigma(\varepsilon_y)}{\sigma(\varepsilon_x)}$, are listed in Table 1 when the excitation signal voltage is at 0.275 V.

Table 1 shows that the value of r is less than 2.5 for the normal pattern and larger than 3.5 for the damage pattern at sensor node 1. At sensor node 2, the value of r is less than 2 for the normal pattern and greater than 5 for the damage pattern. The acceleration signals at sensor nodes 1 and 2 for both normal pattern and simulated damage pattern are shown in Figures 10 and 11. The amplitude of the damage acceleration signals are smaller compared to the normal acceleration signals.

5.4 Mobile agents for structural damage diagnosis

For the mobile agent test, a user laptop sends mobile agent 1 to the sensor node 1 and mobile agent 2 to the sensor node 2. The task of each mobile agent is to diagnose structural damage using AR-ARX method

Table 1
Off-line calculation of the ratio of standard deviation of the residual time series

Sensor	*Pattern*	*Ratio* $(\sigma(\varepsilon_y)/\sigma(\varepsilon_x))$							
1	Normal	1.5240	2.0204	1.6099	1.6241	2.4093	1.8547	1.5547	1.8743
		2.0055	1.2815	1.5093	1.7076	1.7283	1.1498	1.8325	1.9087
	Damage	3.8485	3.8225	4.2859	4.0668	3.8307	3.8489	4.1123	3.8565
		3.8528	3.8592	4.1418	4.1415	3.8429	4.1251	4.1180	4.1094
		4.1293	4.1176	3.8475					
2	Normal	1.5054	1.0246	1.5293	1.4173	1.7241	1.6026	1.9286	1.6165
		1.5583	1.1919	1.3007	1.4971	1.1587	1.2641	1.4014	1.3650
	Damage	7.5001	14.4038	7.6681	14.3588	7.7951	7.5751	7.2733	
		6.9559	12.7211	7.0272	5.9883	5.9110	7.2269	12.5588	
		6.3172	10.9039	6.2257	6.1708	6.0201			

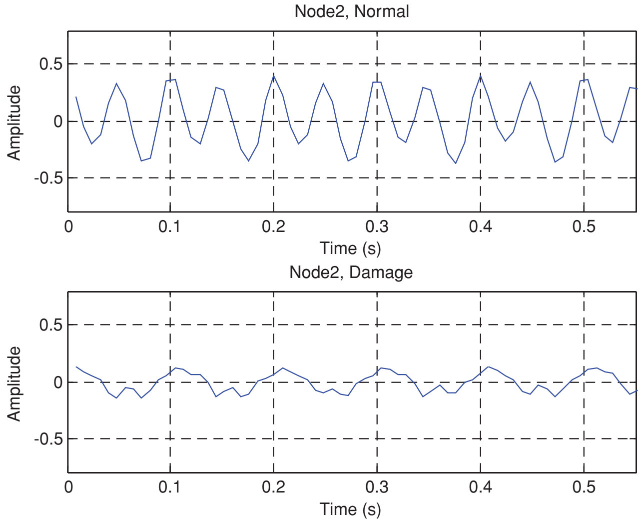

Fig. 11. Sensor node 2 acceleration signals.

based on the acceleration data collected by each sensor node. Mobile agents migrate via mobile agent messages. A mobile agent message in Mobile-C is represented in XML format, and contains general information of a mobile agent and tasks that the mobile agent is going to perform on destination hosts (Chen et al., 2008b). The general information of a mobile agent includes agent name, agent owner, and the home of the agent. Task information includes number of tasks, a task progress pointer, and the definition for each task such as the hosts to perform tasks, return variables, and the agent code for each task. A mobile agent can visit a number of hosts. In our example, only one destination host is assigned to each mobile agent.

Mobile agent code is a regular C program. In this example, the flowchart of the mobile agent code is shown in Figure 12. A mobile agent reads the acceleration data on the residing sensor node and calculates the coefficients of the AR model. Based on the calculated AR coefficients, the mobile agent searches an AR model in the AR-ARX reference file on the sensor node, which has the smallest Euclidean distance with the calculated AR model. Once the matched AR model is found, the coefficients of the ARX model paired with the matched AR model $[\alpha_i(i = 1, 2, 3, 4, 5), \beta_j(j = 1, 2, 3, 4, 5)]$ will be used to calculate the standard deviation of the residual errors of the measurement data $\sigma(\varepsilon_y)$ and determines if damage presents based on the ratio of $\frac{\sigma(\varepsilon_y)}{\sigma(\varepsilon_x)}$. The value of $\sigma(\varepsilon_x)$ is the standard deviation of the residual errors of the matched ARX model in the reference file. If the value of $r = \frac{\sigma(\varepsilon_y)}{\sigma(\varepsilon_x)}$ is greater than a predefined threshold, a structural damage presents. Otherwise, no damage presents.

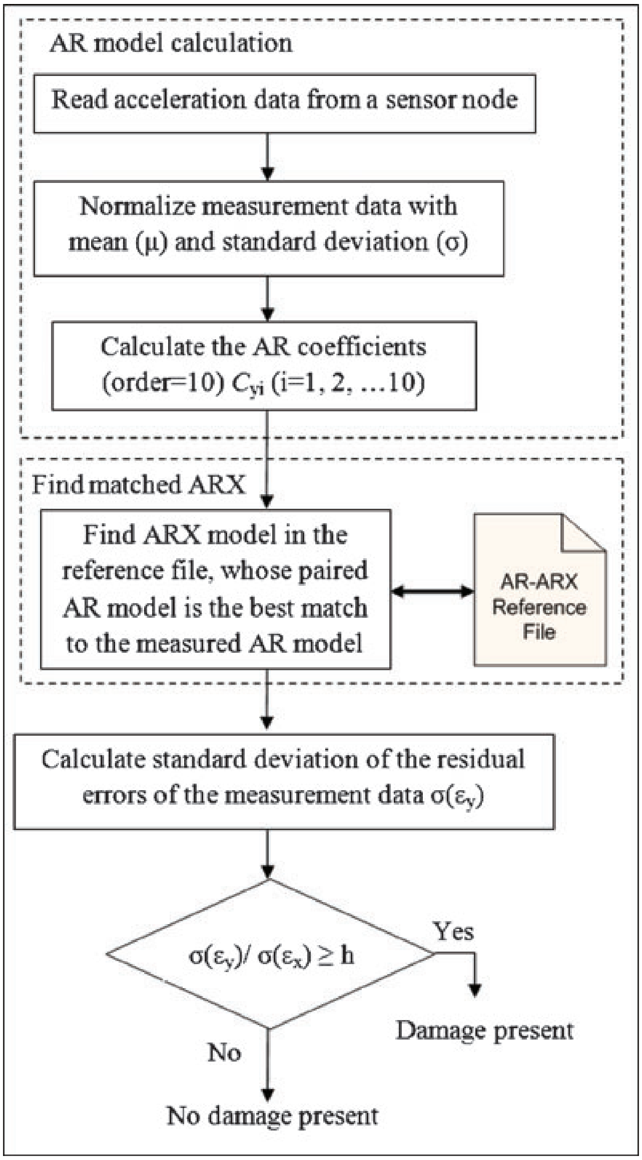

Fig. 12. The flowchart of the mobile agent code.

During simulated damage test, each sensor node ran mobile agent server program waiting for mobile agents. When a mobile agent arrived at a sensor node, it performed structural damage diagnosis using equipped AR-ARX algorithm and the acceleration data collected by the local sensor node. The execution of the AR-ARX algorithm was supported by the execution engine of the mobile agent system installed on the sensor node. Based on the off-line calculation results shown in Table 1, the threshold of the standard deviation ratio was selected to be 3.5 for the sensor node 1 and 5.0 for the sensor node 2, respectively. The calculated ratios of standard deviation were 3.931253 at the sensor node 1 and 6.284010 at the sensor node 2. Both ratios were greater than the threshold selected for the sensor nodes 1 and 2. As a result, structural damage was detected at both sensor nodes 1 and 2. The damage diagnosis results of both mobile agents were sent back to the mobile agent dispatcher and displayed on its terminal.

6 DISCUSSIONS

The strength of the mobile agent approach in enhancing flexibility and reducing data transmission has been demonstrated in the previous sections. Damage diagnosis algorithms originally not designed in the sensor nodes, such as AR-ARX in the given example, can be dynamically added to the sensor nodes during the monitoring process without the need to interrupt normal operation. This feature provides great flexibility for an SHM sensor network to adopt newly developed diagnosis algorithms and change monitoring tasks. In addition, the local damage diagnosis does not require transmitting sensor data to a central data station. The reduction of data transmission is significant in SHM sensor networks comparing to environmental monitoring networks as SHM monitoring networks usually have a high sampling frequency. The typical accelerometer sampling rate for SHM is between 100 to 150 sps. We chose 125 sps in the given example. Each acceleration reading includes two bytes of data. For the acceleration measurement, the sensor node collects data from a 3-axis accelerometer. The sensor node generates $3 \times 125 \times 2 = 750$ bytes per second (45,000 byte/min or 2.7 MB/h). The size of the agent message, on the other hand, is 8,722 bytes. Once a mobile agent is dispatched, it can work independently at the remote sensor nodes to perform assigned tasks.

Mobile agents, in theory, can deploy any algorithm programs on remote sensors. There are several practical issues, however, we would like to discuss in this section. Structural damage diagnosis needs two major components: structural dynamic response data and diagnosis algorithms. The SHM algorithms typically involve intensive numerical computation for data analysis and decision making. The size of mobile agent messages and the diagnosis speed depend on the availability of numerical functions at sensor nodes. If all the necessary numerical functions are available at sensor nodes, the size of a mobile agent message will be reduced and the speed of the damage diagnosis will be increased. The reduction of the mobile agent size is obvious because the mobile agent does not need to carry on required numerical function code. For the diagnosis speed, the mobile agent code is typically executed interpretively, which is slower in comparison to executing binary code. If numerical functions are integrated into binary libraries at sensor nodes, the small mobile agent code and high diagnosis speed can be achieved.

The example given in the article is a completely distributed algorithm that only needs acceleration data from the local sensor node. For algorithms that extract damage information from multiple sensors, the time-synchronized data are needed. In this case, a feasible solution is to have a mobile diagnosis agent migrate to the cluster head of a subnetwork that covers the area of interest. The mobile diagnosis agent collaborates with data acquisition agents (typically stationary agents) residing in sensor nodes in the region to perform a synchronized data acquisition. Time-synchronized data acquisition for wireless sensor networks has been investigated extensively. A number of synchronization schemes (Elson et al., 2002; Ganeriwal et al., 2003; Van Greunen and Rabaey, 2003; Xu et al., 2004) are proposed to achieve network-wide synchronization.

7 CONCLUSIONS

This article presents a mobile agent approach to reduce data transmission and enhance the flexibility of wireless SHM sensor networks. Wireless sensor networks are bandwidth constrained. In such a system, the centralized management and data processing is challenging due to the frequent communication among network components and data transmission. This is especially severe for a wireless sensor network with a high sampling rate, such as structural health monitoring networks. The presented mobile agent approach distributes damage diagnosis algorithms, such as AR-ARX algorithm, to the sensor nodes instead of transmitting sensor data to a central data station for the damage diagnosis. The data processing and damage diagnosis are performed at sensor nodes. Compared to transmitting sensor data to a central station, transporting algorithm code significantly reduces the traffic load over a sensor network. In addition, the ability to dynamically deploy diagnosis algorithms and control strategies on sensor nodes allows an SHM sensor network to make appropriate adjustments in response to operational and task changes. The validation example given in the article shows that the presented mobile agent approach can successfully deploy AR-ARX damage diagnosis algorithm on distributed sensor nodes via mobile agents. The acceleration data processing and damage diagnosis are performed by mobile agents at local sensor nodes.

ACKNOWLEDGMENTS

The authors would like to thank Dr. Chuanzhi Zang and undergraduate research assistant Justin Slepak, in the

Laboratory of Intelligent Mechatronic and Embedded Systems at Michigan Technological University for their contributions to the presented work.

REFERENCES

Adeli, H. & Jiang, X. M. (2006), Dynamic fuzzy wavelet neural network model for structural system identification, *Journal of Structural Engineering ASCE*, **132**(1), 102–11.

Anderson, E., Bai, Z., Bischof, C., Blackford, S., Demmel, J., Dongarra, J., Croz, J. D., Greenbaum, A., Hammarling, S., McKenney, A. & Sorensen, D. (1999), LAPACK users' guide, Society for Industrial and Applied Mathematics, Philadelphia, PA.

Blumenthal, J., Handy, M., Golatowski, F., Haase, M. & Timmermann, D. (2003), Wireless sensor networks—new challenges in software engineering, *EFTA 2003. 2003 IEEE Conference on Emerging Technologies and Factory Automation*. Proceedings (Cat. No. 03TH8696), Lisbon, Portugal.

Boulis, A., Han, C.-C. & Srivastava, M. B. (2003), Design and implementation of a framework for efficient and programmable sensor networks, in *Proceedings of the 1st International Conference on Mobile Systems, Applications and Services MobiSys '03 San Francisco*, California, CA.

Carden, E. P. & Brownjohn, J. M. W. (2008), Fuzzy clustering of stability diagrams for vibration-based structural health monitoring, *Computer-Aided Civil and Infrastructure Engineering*, **23**(5), 360–72.

Ch (2009), Ch—An Embeddable C/C++Interpreter, http://www.softintegration.com.

Chen, B. (2008), Mobile agent technology for distributed sensing and actuation systems. *ASME 2008 Dynamic Systems and Control Conference*, Ann Arbor, MI.

Chen, B., Cheng, H. H. & Palen, J. (2006), Mobile-C: A mobile agent platform for mobile C/C++ agents, *Software-Practice & Experience*, **36**(15), 1711–33.

Chen, B., Cheng, H. H. & Palen, J. (2009), Integrating mobile agent technology with multi-agent systems for distributed traffic detection and management systems, *Transportation Research Part C: Emerging Technologies*, **17**(1), 1–10.

Chen, H. C., Goldberg, M., Magdon-Ismail, M. & Wallace, W. A. (2008a), Reverse engineering a social agent-based hidden Markov model - ViSAGE, *International Journal of Neural Systems*, **18**(6), 491–526.

Chen, B., Linz, D. D. & Cheng, H. H. (2008b), XML-based agent communication, migration and computation in mobile agent systems, *Journal of Systems and Software*, **81**(8), 1364–76.

Chen, B. & Zang, C. (2009), Artificial immune pattern recognition for structure damage classification, *Computers & Structures*, **87**(21–22), 1394–1407.

Doebling, S. W., Farrar, C. R., Prime, M. B. & Shevitz, D. W. (1996), *Damage Identification and Health Monitoring of Structural and Mechanical Systems from Changes in Their Vibration Characteristics: A Literature Review*, Los Alamos National Laboratory, Los Alamos, NM.

Elson, J., Girod, L. & Estrin, D. (2002), Fine-grained network time synchronization using reference broadcasts, in *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation (OSDI 2002)*, Boston, MA.

Farrar, C. R., Allen, D. W., Park, G., Ball, S. & Masquelier, M. P. (2006b), Coupling sensing hardware with data interrogation software for structural health monitoring, *Shock and Vibration*, **13**(4–5), 519–30.

Farrar, C. R., Park, G., Allen, D. W. & Todd, M. D. (2006a), Sensor network paradigms for structural health monitoring, *Structural Control & Health Monitoring*, **13**(1), 210–25.

FIPA (2009), The Foundation for Intelligent Physical Agents. Available at: http://www.fipa.org/.

Ganeriwal, S., Kumar, R. & Srivastava, M. B. (2003), Timing-sync protocol for sensor networks, in *Proceedings of the First International Conference on Embedded Networked Sensor Systems*, ACM Press, Los Angeles, CA.

Gao, Y., Spencer, B. F. & Ruiz-Sandoval, M. (2006), Distributed computing strategy for structural health monitoring, *Structural Control & Health Monitoring*, **13**(1), 488–507.

Gumstix (2009), Gumstix Inc. Available at: http://www.gumstix.com.

Hadim, S. & Mohamed, N. (2006a), Middleware: Middleware challenges and approaches for wireless sensor networks, *IEEE Distributed Systems Online*, **7**(3), 1–23.

Hadim, S. & Mohamed, N. (2006b), Middleware for wireless sensor networks: A survey, *First International Conference on Communication System Software and Middleware*, Comsware, Delhi, India.

He, X. F., Moaveni, B., Conte, J. P. & Elgamal, A. (2008), Modal identification study of Vincent Thomas Bridge using simulated wind-induced ambient vibration data, *Computer-Aided Civil and Infrastructure Engineering*, **23**(5), 373–88.

Jiang, X. & Adeli, H. (2005), Dynamic wavelet neural network for nonlinear identification of highrise buildings, *Computer-Aided Civil and Infrastructure Engineering*, **20**(5), 316–30.

Jiang, X. & Adeli, H. (2007), Pseudospectra, MUSIC, and dynamic wavelet neural network for damage detection of highrise buildings, *International Journal for Numerical Methods in Engineering*, **71**(5), 606–29.

Jiang, X. & Adeli, H. (2008a), Neuro-genetic algorithm for non-linear active control of structures, *International Journal for Numerical Methods in Engineering*, **75**(7), 770–86.

Jiang, X. & Adeli, H. (2008b), Dynamic fuzzy wavelet neuroemulator for non-linear control of irregular building structures, *International Journal for Numerical Methods in Engineering*, **74**(7), 1045–66.

Jiang, X., Mahadevan, S. & Adeli, H. (2007), Bayesian wavelet packet denoising for structural system identification, *Structural Control and Health Monitoring*, **14**(2), 333–56.

Kolakowski, P. (2007), Structural health monitoring—a review with the emphasis on low-frequency methods, *Engineering Transactions, IPPT*, **55**(3), 239–75.

Levis, P. & Culler, D. (2002), Mate: A tiny virtual machine for sensor networks, *ACM SIGPLAN Notices*, **37**(10), 85–95.

Li, S. & Wu, Z. (2008), A model-free method for damage locating and quantifying in a beam-like structure based on dynamic distributed strain measurements, *Computer-Aided Civil and Infrastructure Engineering*, **23**(5), 404–13.

López-París, D. & Brazález-Guerra, A. (2009), A new autonomous agent approach for the simulation of pedestrians in urban environments, *Integrated Computer-Aided Engineering*, **16**(4), 283–97.

Lynch, J. P., Law, K. H., Kiremidjian, A. S., Carryer, E., Kenny, T. W., Partridge, A. & Sundararajan, A. (2002), Validation of a wireless modular monitoring system for structures, *Smart Structures and Materials 2002: Smart Systems for Bridges, Structures, and Highways*, **4696**, 124–35.

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

Moaveni, B., He, X., Conte, J. P. & de Callafon, R. A. (2008), Damage identification of a composite beam using finite element model updating, *Computer-Aided Civil and Infrastructure Engineering*, **23**(5), 339–59.

Monticolo, D., Hilaire, V., Gomes, S. & Koukam, A. (2008), A multi-agent system for building project memories to facilitate the design process, *Integrated Computer-Aided Engineering*, **15**(1), 3–20.

Nagayama, T. & Spencer, B. F., Jr. (2007), Structural health monitoring using smart sensors, *Newmark Structural Engineering Laboratory Report Series 001*, Urbana, IL.

Nagayama, T., Spencer, B. F., Jr. & Rice, J. A. (2009), Autonomous decentralized structural health monitoring using smart sensors, *Structural Control & Health Monitoring*, **16**(7–8), 842–59.

Pakrashi, V., O'Connor, A. & Basu, B. (2007), A study on the effects of damage models and wavelet bases for damage identification and calibration in beams, *Computer-Aided Civil and Infrastructure Engineering*, **22**, 555–69.

Park, H. S., Lee, H. M., Adeli, H. & Lee, I. (2007), A new approach for health monitoring of structures: Terrestrial laser scanning, *Computer-Aided Civil and Infrastructure Engineering*, **22**(1), 19–30.

Press, W. H., Flannery, B. P., Teukolsky, S. A. & Vetterling, W. T. (1992), *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge University Press, Cambridge, UK.

Psimoulis, P. A. & Stiros, S. C. (2008), Experimental assessment of the accuracy of GPS and RTS for the determination of the parameters of oscillation of major structures, *Computer-Aided Civil and Infrastructure Engineering*, **23**(5), 389–403.

Quickfilter (2008), QF4A512 4-Channel Programmable Signal Converter. Available at http://www.quickfiltertech.com/.

Rice, J. A. & Spencer, B. F., Jr. (2008), Structural health monitoring sensor development for the Imote2 platform, *Proceedings of the SPIE—The International Society for Optical Engineering*, **6932**, 693234–1–12, San Diego, CA.

Römer, K. (2004), Programming Paradigms and Middleware for Sensor Networks. Available at: http://www.citeseer.ist.psu.edu/666689.html.

Sohn, H. & Farrar, C. R. (2001), Damage diagnosis using time series analysis of vibration signals, *Smart Materials and Structures*, **10**(3), 446–51.

Sohn, H., Farrar, C. R., Hemez, F. M., Czarnecki, J. J., Shunk, D. D., Stinemates, D. W. & Nadler, B. R. (2003), A review of structural health monitoring literature: 1996–2001, Los Alamos National Laboratory Report, Los Alamos, NM.

Sohn, H., Kim, S. D. & Harries, K. (2008), Reference-free damage classification based on cluster analysis, *Computer-Aided Civil and Infrastructure Engineering*, **23**(5), 324–38.

Soyoz, S. & Feng, M. Q. (2009), Long-term monitoring and identification of bridge structural parameters, *Computer-Aided Civil and Infrastructure Engineering*, **24**(2), 82–92.

Spencer, B., Ruiz-Sandoval, M. & Kurata, N. (2004), Smart sensing technology: Opportunities and challenges, *Structural Control and Health Monitoring*, **11**(4), 349–68.

Su, H. Z., Wu, Z. R. & Wen, Z. P. (2007), Identification model for dam behavior based on wavelet network, *Computer-Aided Civil and Infrastructure Engineering*, **22**(6), 438–48.

Szumel, L., LeBrun, J. & Owens, J. D. (2005), Towards a mobile agent framework for sensor networks, *The Second IEEE Workshop on Embedded Networked Sensors (IEEE Cat. No. 05EX1105)*, Sydney, Australia.

Tanner, N. A., Wait, J. R., Farrar, C. R. & Sohn, H. (2003), Structural health monitoring using modular wireless sensors, *Journal of Intelligent Material Systems and Structures*, **14**(1), 43–56.

Van Greunen, J. & Rabaey, J. (2003), Lightweight time synchronization for sensor networks, in *Proceedings WSNA*, San Diego, CA.

Xu, N., Rangwala, S., Chintalapudi, K. K., Ganesan, D., Broad, A., Govindan, R. & Estrin, D. (2004), A wireless sensor network for structural monitoring, *SenSys '04: Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, Baltimore, MD.

Yu, Y., Krishnamachari, B. & Prasanna, V. K. (2004), Issues in designing middleware for wireless sensor networks, *IEEE Network*, **18**(1), 15–21.