SnowFort: An Open Source Wireless Sensor Network for Data Analytics in Infrastructure and Environmental Monitoring

Yizheng Liao, *Student Member, IEEE,* Mark Mollineaux, Richard Hsu, Rebekah Bartlett, Anubhav Singla, Adnan Raja, Ravneet Bajwa, and Ram Rajagopal, *Member, IEEE*

***Abstract*—SnowFort, an open source wireless sensor network (WSN) for data analytics, is proposed for monitoring infrastructure and environment. The wireless sensing unit is optimized to be low power for extremely long-term deployments. Several features, such as data compression and online reconfiguration, are introduced to further reduce power consumption. A low power wireless sensor network over optimized time division multiple access (TDMA) scheme is designed to be scalable and reliable for a network with hundreds of sensors. Real-time data visualization and analytical tools are provided with a representational state transfer (RESTful) application programming interface (API). We utilize SnowFort to develop a real-time damage detection application in structural health monitoring. We develop a distributed algorithm robust to data loss and validate it in a laboratory setup.**

Keywords*—*sensor system integration, structural health monitoring, wireless sensor networks, data analytics, sensing platform.

I. Introduction

WIRELESS sensor networks (WSNs) for infrastructure and environmental monitoring has evolved over several generations. Various successful efforts to design and develop wireless monitoring systems include [1]–[3]. Comprehensive summaries and discussions of system prototypes on the wireless monitoring system are given in [4]. The wireless monitoring system can be applied to buildings [5], [6], habitat [7], [8], bridges [9], traffic [10], pipelines [11], volcanoes [12], and environment [13]. The main components of a wireless monitoring system include wireless sensing units, wireless communication networks, and decision support systems. Existing solutions and deployments have focused on specific components of the system and in general were designed for utilization by WSN experts. Large scale adoption of WSN technology requires a platform that addresses several challenges [14]:

- Durability: sensing units need to function for months to years without battery replacement.
- Reliability: sensors need to reliably deliver data over imperfect wireless communication channels.
- Adaptability: dynamically adding and removing sensors to the system should be simple.
- Intelligence: scalable real-time validation, analysis and processing of the data needs to be available.
- Simplicity: customizing, installing, deploying and maintaining the system should be simple.

SnowFort, Sensor Network: Open & Wireless for daTa analytics, is proposed in this paper [15]. SnowFort is an open source wireless sensor system designed for infrastructure and environmental monitoring. It simplifies scalable implementation and deployment of WSN to collect, cleanse, analyze and visualize infrastructure and environmental data in real-time. It also supports a variety of hardware platforms and can be integrated with existing systems.

SnowFort provides an interface that simplifies integration of multiple types of sensors into the wireless unit. The communication scheme is reliable, stable, energy efficient and simple to diagnose. Information processing in real-time is provided by a decision support system (DSS) that relies in recent analytics frameworks. SnowFort is able to support up to 54 sensing units per very low cost base station. The sensing unit powered by two AA

Y. Liao, M. Mollineaux, and R. Rajagopal are with the Department of Civil and Environmental Engineering, Stanford University, Stanford, CA 94305, USA (e-mail: yzliao@stanford.edu; mgm1@stanford.edu; ramr@stanford.edu).

R. Hsu was with the Department of Computer Science, Stanford University, Stanford, CA 94305, USA (e-mail: richardhsu@alumni.stanford.edu).

R. Bartlett is with the Department of Electrical Engineering, Stanford University, Stanford, CA 94305, USA (e-mail: rbartlet@stanford.edu).

A. Singla was with the Department of Electrical Engineering, Stanford University, Stanford, CA 94305, USA (e-mail: anubhavs@alumni.stanford.edu).

A. Raja and R. Bajwa were with the Department of Civil and Environmental Engineering, Stanford University, Stanford, CA 94305, USA (e-mail: adnanster@gmail.com; ronniebajwa@gmail.com).

batteries can last between 125 to 244 days operating an accelerometer and temperature sensor. The DSS supports the real-time analysis and visualization of data from multiple large scale wireless monitoring systems via a web interface.

This paper is organized as follows. Section II discussed related works on wireless monitoring systems. In Section III, the system architecture, including the wireless sensing unit and the decision support system, is discussed. In Section IV, the customized wireless communication scheme and several new features are introduced. In Section V, an application of SnowFort in structural health monitoring is presented. In Section VI, results of the SnowFort system performance and experiment results of structural health monitoring are demonstrated. Section VII draws summaries and conclusions.

II. Related Works

Two well-known WSN data acquisition systems [16] and [17] have power-aware optimization and support large scale networks but lacks analytics intellgence. The SQL-syntax query processor offers significant complications for infrastructure and environmental applications. In addition, the systems are not optimized to address several issues common in infrastructure and environmental applications, such as high sampling frequency and low data synchronization error. SnowFort not only shares the same advantages of [16] and [17] but also addresses these specific problems.

ISHMP system developed at the University of Illinois at Urbana-Champaign is an open-source WSN platform designed for structural monitoring [18]. It has been experimentally validated for damage identification and localization by [19] and [20]. However, ISHMP system lacks DSS, which is a critical component of today's wireless monitoring system and is included in SnowFort. [21] proposes a framework includes all the three components required for wireless monitoring system. Unlike SnowFort, which supports multipe devices and has web interface, this system is a OS-specific executable program. Further, the framework cannot be integrated with any existing system that has been proposed.

The indoor wireless monitoring system in [22] captures the morphology of buildings for rapid development and flexible management. The indoor WSN in [23] focuses on the event logic in building management with multi-level decision system. Both frameworks effectively and efficiently make decisions based on sensing data. However, they do not address network reliability and data synchronization. [24] is an indoor Heating, Ventilation and Air Conditioning (HVAC) system with DSS at the back-end. However, this HVAC system relies on WiFi logs, not sensing units. [13] is a outdoor wireless system for environmental monitoring. It achieves reliable communication and data synchronization by customized MAC and networking protocol, which is similar to SnowFort. This system is validated by experiments and achieved the long-term operation by employing the solar panels. In SnowFort, the wireless sensors are deployed in the field. The reliable wireless communication is employed for data acquisition with power-aware optimization and data synchronization mechanism. The building management system can be achieved on the web server.

A time-domain algorithm to detect damage of a structure via WSN is discussed in [25]. This method extracts damage sensitive features from multiple accelerometers and strain garages and uses hypothesis tests to decide whether damages occur. This algorithm is centralized batch processing. The framework presented in [26] uses damage sensitive features for testing as well, and in a sequential manner. [27] presents a frequency-domain distributed algorithm for the damage detection and is validated by using the ISHMP system. In this paper, we extend our prior work in [26] to a time-domain distributed algorithm with imporved resistance to communication packet missing. In addition, we validate the performances of our distributed algorithm by introducing packet loss, which has not been discussed in [27] or other works.

III. System Architecture

SnowFort is architected to address each requirement of a scalable WSN platform. Performance requirements and typical application constraints in infrastructure and environmental monitoring drive the design choices for the system. Durability and reliability for the system are achieved by careful design of the wireless communication protocol, in particular the medium access (MAC) layer. Simplicity and scalability are achieved by ensuring a simplified single-hop network topology, which is extensible by relying on multiple base-stations. Intelligence is provided relying on high performance computation and web systems to support visualization and data-processing in real-time. The system provides high-level functions and APIs to simplify custom development. SnowFort strives to be transparent, and is shared as an open source project that can enable community development of a stable, durable, reliable, adaptable and simple solution for infrastructure and environmental monitoring.

The system architecture of SnowFort has four parts (Fig. 1): a wireless sensor mote, a base station, a cloud server, and a web interface. The wireless sensor mote is the sensing unit. The base station or access point intermediates a group of motes to form a network. The cloud server and the web interface compose the decision support system. The remainder of the section details the functionality and design choices of each component.

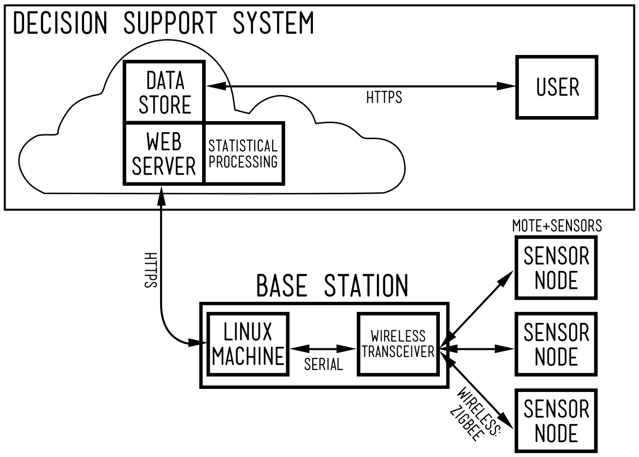

Fig. 1. System architecture of SnowFort.

A. Wireless Sensor Mote

A wireless sensor mote is a battery-powered sensor system with microprocessor, wireless transceiver, and memory. It collects data, compresses and transmits them to a base station. Also, the mote receives and executes commands from the base station. The mote has analog-to-digital converters (ADC), digital-to-analog converters (DAC), and inter-integrated circuit (I^2C) ports, which connect and communicate with microelectromechanical sensors (MEMS), such as accelerometer, gyroscope, and temperature sensor. A typical mote is shown in Fig. 2. SnowFort utilizes Contiki as the embedded operating system (OS) for motes. Contiki is an open source OS written in C [28]. It facilitates low level software development by providing high-level routines to interface with a variety of motes. It was chosen due to its simplicity, lightweight code base, and support for multiple hardware platforms. SnowFort modifies and extends Contiki in several ways. We implement a simpler networking protocol and include low level sensor functionality such as data compression and remote reconfiguration. More details are given in Section IV.

B. Base Station

In SnowFort, the primary roles of the base station are coordinating communications within the network, receiving data from motes, and transmitting data to the cloud server via the Internet.

The base station, which is typically the single component connected to a power source, runs in a low cost Linux machine (Raspberry Pi) and includes a CC2420 transceiver. Also, the base station has capability to access the Internet with high data rates via Ethernet, cellular network, sattllelte network, or WLAN. It executes a modifiable Python script to implement the communication protocol to the motes, decompress the data received from motes and transmit to the cloud server. Simple statistics and local storage functionality are included as well but processing capabilities are kept lightweight.

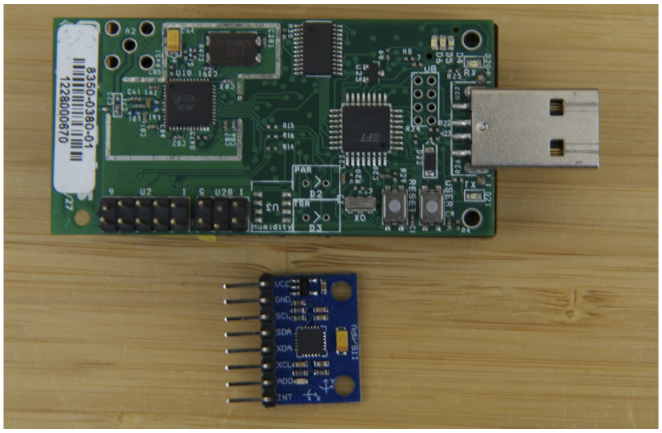

Fig. 2. The green device (upper) is the Telosb mote with TI MSP430 microprocessor and CC2420 transceiver [29]. The blue device (lower) is the MPU 6050 I^2C sensor with one three-axis accelerometer, one three-axis gyroscope, and one temperature sensor [30].

C. Cloud Server

In infrastructure and environmental monitoring applications, the decision support system (DSS) is an important element [21]. Deploying large networks requires a scalable and robust DSS built to handle massive data flows, real-time processing and visualization. In SnowFort, the cloud server is the primary data storage and processing unit. It also serves as the web server for the front-end user interface. The cloud server has three components: a web server, a data storage, and a statistical data processing unit.

The web server receives data posted from the base station and handles requests from end-users. In addition, the web server sends commands from end-users to the base station. Multiple base stations can simultaneously post to the server. In SnowFort, the web framework is developed in Flask, an open source web framework in Python. The data communication protocol between the base station and the web server is the HTTPS protocol. The web server also supports a JSON data format and a representational state transfer (RESTful) API for data communication between systems. Therefore, any WSN which adheres to this API and data format can use the SnowFort cloud server and web interface for data visualization and analytics. This standardization extends SnowFort visualization and analytic tools to any existing networks beyond SnowFort motes. A standard authentication protocol is implemented to prevent access from unauthenticated users.

The data storage unit has two databases for storing data. The first database stores raw data in real-time. The second database stores cleansed real-time data which can be used for statistical modeling. The web server inserts the formatted data into the appropriate database by calling a standard API. Currently, both databases are implemented in MySQL and the database API is provided by the MySQL Python library. When datasets scale to large sizes, it is desirable to utilize distributed database systems such as Apache Hadoop, Apache Hive, or Apache Spark. The database updating functions in SnowFort can be substituted by the standard APIs of these systems. Therefore, the MySQL databases can be easily replaced by various commonly used distributed database systems. In addition, SnowFort provides interfaces for scientific computation softwares, such as R and Matlab, to access data. Such feature will benefit the applications like sensor placement and optimization ([31], [32]).

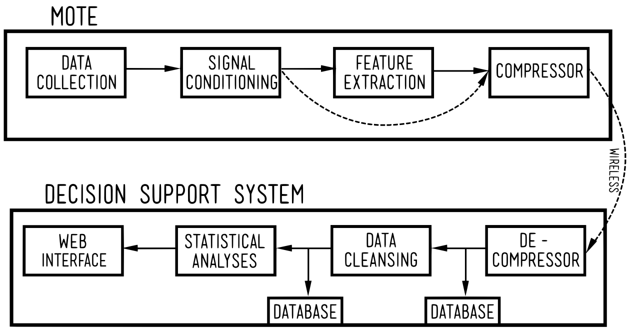

Fig. 3. Data processing components in SnowFort

The data processing component in SnowFort is shown in Fig. 3. Data samples collected by the mote are conditioned (e.g filtering) and features from the signal are extracted. Typical features include the raw measurements themselves, statistical means, variances and autoregressive coefficients as shown in Section V. These data are then compressed and sent by the mote to the base station. The base station forwards the decompressed data to the cloud server. The cloud server stores the data in the raw data table. Data cleansing algorithms are applied and the results stored in the second database. Different motes can have different sampling rates and values might be missing due to dropped packets. Data cleansing consists of verifying the integrity of the data with simple consistency checks and normalizing the data to a canonical form by filling in missing values and resampling into a common sampling basis. Finally, application dependent statistical analysis are applied and results can be visualized in the web interface. The analysis can be done by batch or real-time processing. The event logics and the event respones can be implemented here as well. Statistical processing is implemented utilizing Python numerical computation and statistical analysis libraries such as NumPy and SciPy.

D. Web Interface

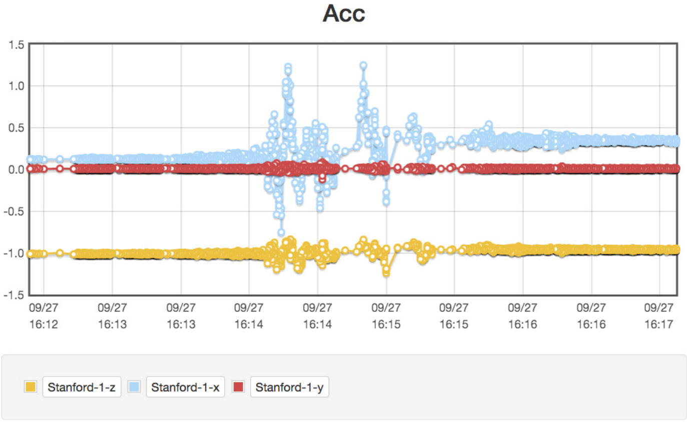

Fig. 4. Data visualization of accelerometer data via SnowFort's web interface. The real-time data from three axes are shown.

SnowFort uses a web interface as the front panel of the decision support system, unlike earlier systems that opted for desktop softwares [21], [33]. The system can be accessed from different locations and a variety of devices. The web interface supports system configuration, data visualization, and user control. Fig. 4 shows a screenshot of real-time data visualization of the web interface. This data visualization panel allows users to view data of multiple sensors from different base stations in real time, view historical data with different time segements, and check each sample value. Also, on this panel, users are able to select the data sources, set basic alarm rules, and perform basic filtering. The system and motes can be remotely managed from the web interface, including rebooting individual motes, changing sampling frequencies, setting mote IDs and configuring the communication system. More configuration details are discussed in Section IV-B.

IV. Firmware

The SnowFort platform is optimized for infrastructure and environmental monitoring applications by directly addressing power consumption, stability and reliability of the network. This section describes the system design details to support these goals.

A. Wireless Communication Scheme

In many current WSN, Zigbee is adopted as the wireless communication protocol. Zigbee uses IEEE 802.15.4 standard radio as the hardware and carrier sense multiple access/collision avoidance (CSMA/CA) as the channel access protocol. Recent literature demonstrates that packet collision rate of CSMA/CA is high in a real-time WSN [34]. Additionally, retransmissions due to collisions consume significant power and result in unpredictable data reception delay. Medium access protocols for WSN have been widely investigated (e.g [35]–[38]) and a comprehensive surveys are presented in [39] and [40]. Time Division Multiple Access-based (TDMA-based) schemes have been demonstrated to be among the most reliable protocols ([10], [41]–[44]).

SnowFort implements a simple TDMA protocol to coordinate the communication between base station and deployed motes. The implementation also provides synchronization functionality. The protocol assumes data is acquired and sent periodically as in most applications in infrastructure and environmental monitoring. The system operates as follows. Time is divided into multiple frames and each frame is further divided into time slots. The first time slot is used by the base station to broadcast a beacon frame, which synchronizes clock and sends commands to the motes. Each mote is assigned a unique and static time slot. The total number of time slots and the length of each time slot are fixed during network configuration. Each mote synchronizes its internal clock upon reception of the beacon frame. It then turns off its radio and schedules the next data transmission based on its assigned time slot. During each time slot only a single mote transmits information while others remain silent so the network is collision free. Motes respond to failure to receive a beacon over a significant period of time by turning off its radio and remaining silent. After some pre-specified time the mote turns on the radio to listen to beacon frames again. The procedure is repeated until communication is reconnected. A summary of the TDMA protocol implementation in SnowFort is shown in Fig. 5. Some features of the implemented scheme are low data packet loss, high performance synchronization (within ms accuracy), significant power savings and fixed and known delay. The low packet loss enables disabling packet acknowledgement for significant power savings.

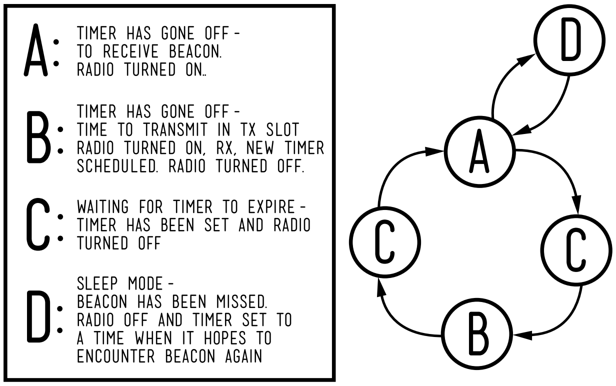
A: TIMER HAS GONE OFF - TO RECEIVE BEACON. RADIO TURNED ON.
B: TIMER HAS GONE OFF - TIME TO TRANSMIT IN TX SLOT RADIO TURNED ON, RX, NEW TIMER SCHEDULED. RADIO TURNED OFF.
C: WAITING FOR TIMER TO EXPIRE - TIMER HAS BEEN SET AND RADIO TURNED OFF
D: SLEEP MODE - BEACON HAS BEEN MISSED. RADIO OFF AND TIMER SET TO A TIME WHEN IT HOPES TO ENCOUNTER BEACON AGAIN
D
A
C
C
B

Fig. 5. State Machine of SnowFort TDMA Protocol.

Many System-on-Chip (SoC) solutions for WSN are designed for the IEEE 802.15.4 standard. These chips also include hardware filters to avoid unnecessary packets on the band of 2.4 GHz. In order to extend SnowFort's TDMA protocol to existing hardware platforms, a customized IEEE 802.15.4 frame header is employed. In SnowFort, the data frame format, one of the four supported frame formats in IEEE 802.15.4, is used for all types of communication. Since the typical maximum number of motes that can be supported by a single network is usually less than 128, PAN identification compression mode and IEEE short address mode are enabled. Thus, the total length of the MAC header is 9 bytes. The total payload size is up to 118 bytes.

By design, the sensor network in SnowFort is organized in a star network topology. Each mote can only communicate with the base station. Therefore, no routing algorithm is required. The network layer is removed from the SnowFort communication stack (Fig. 6). The application layer performs the sensor sampling task. The sampled data are pushed to the MAC layer for transmission directly. The MAC address of the mote serves as the unique identification in the network. When the mote starts to transmit, sampled data are included as the payload of the MAC frame. This simplification reduces execution cycles of motes and saves battery power. TDMA and the star network topology limit the communication range of the motes to the maximum range of the radio devices, thus limiting the scalability of the network. SnowFort addresses this limitation by

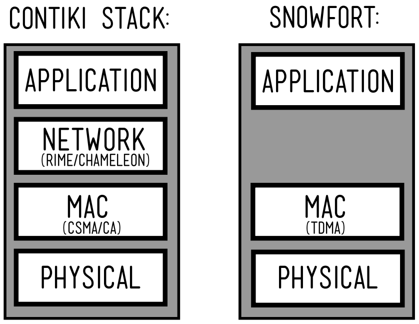

Fig. 6. Contiki network stack and SnowFort network stack.

adding spatially spread base stations to the network. Base stations are very low cost so can be freely deployed. Frequency division multiple access is used to enable all base stations to operate simultaneously. Base stations covering adjacent spatial regions are assigned different RF channels for channel access, following traditional ideas from 2G cellular networks [45]. The base stations are sychronized by using GPS or the network time protocol (NTP).

B. Remote Shell

In Contiki, a command line interface, or shell, is enabled for executing commands from users. Users can connect the mote via physical ports, such as USB and UART, and send commands via standard input. This feature helps them configure the mote while the system is operating. However, in most WSN applications, motes are difficult or costly to access after installation and deployment. For example, in [10], a WSN for vehicle classification was deployed. In this network, sensors were embedded inside the pavement and cannot be accessed physically after installation. SnowFort implements a remote shell to enable configuration after installation.

The remote shell can receive and execute commands from users via the wireless connections. Users can utilize the web interface to send various configuration commands to the network as shown in Section III. The command is then forwarded to the base station. The command with parameters is included as the payload of a beacon frame. The command can be transmitted to one mote, a set of motes, or all motes. Motes deployed in the field check the payload of the beacon frame and decide whether to execute the commands or not. Table I shows some of the available commands in SnowFort.

TABLE I. COMMANDS SUPPORTED IN SNOWFORT

Command	Command Function
nodeid *id*	set mote id to *id*
txpower *power*	set transmission power to *power* (0 - 31)
rfchannel *channel*	set radio channel to *channel* (11 - 26)
timeslot *ts*	set TDMA time slot to *ts*
reboot	reboot mote in 4 seconds
blink *num*	blink LEDs *num* times

C. Data Compression

In WSN, the radio transmission is one of the most power consuming components. For the Telosb mote, the power consumption of transmitting one bit is equivalent to approximately that of 350 CPU execution cycles. SnowFort implements a standard lossless data compression algorithm ([46]) to reduce data payloads. The mechanism is summarized next. Let $x[i]$ denote the binary representation of the R bit data sample i. The sample residual is defined as $d[i] = x[i] - x[i-1]$ (for $d[0]$, let $x[-1]$ be the quantized value of the median of the measurement range). The compressor encodes the residual $d[i]$ based on its probability density distribution. It outputs a binary sequence $b[i] = (c[i], a[i])$. Here $c[i]$ is a Huffman variable length code with $n[i]$ bits used to represent $d[i]$. $a[i]$ denotes a truncated binary representation of $d[i]$ and (p, q) denotes the concatenation of p and q. When $d[i]$ is 0, $n[i] = 0$ and $a[i]$ is not included in $b[i]$. When $d[i] > 0$, $a[i] = (d[i])|_{n[i]}$, where $v|_n$ denotes the n least significant bits of v. When $d[i] < 0$, $a[i] = (d[i] - 1)|_{n[i]}$. Compression works because typically data has temporal correlation and the residual requires less bits to represent it than the original data. The decompressor is implemented in the Linux machine and uses a lookup table to decode messages. As shown in [46], this method can be applied to highly-correlated data samples as well.

[46] provides Huffman variable length codes for temperature and relative humidity data. These two types of data are slowly varying in real applications, and the corresponding residuals exhibit zero-mean approximately Gaussian distributions. This is not the case for accelerometer and strain gauge data, although they are highly correlated in consecutive samples. Fig. 7 displays probability density functions of typical accelerometer and strain gauge data from SHM experiments [25]. The residuals are not zero mean and not Gaussian. SnowFort learns a codebook for each type of sensor data based on the actual observed statistics of residuals, so each type of data has a unique Huffman variable length codebook. Compression observed in practice for the mechanism ranges from none to 64.68%.

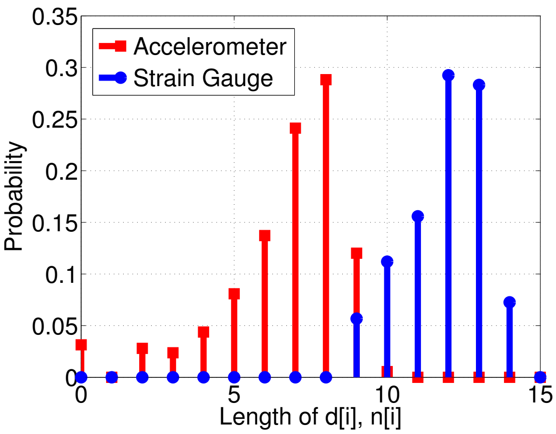

Fig. 7. Probability density functions of data residuals for acceleration and strain with $R = 15$.

V. Structural Health Monitoring Application

An important application of wireless infrastructure monitoring is structural health monitoring (SHM). Significant progress in the utilization of WSN in SHM has been made [4], [25], [47]. The initial focus was on identifying *features* of strain gauge and accelerometer measurements that were sensitive to damage. A feature is a function extracted from samples of data. Pattern based damage detection in particular focused on developing features based on methods such as wavelets and AR model coefficients [4]. These features are called Damage Sensitive Features (DSFs). Sequential optimal decision making utilizing features has recently been developed that fuses information from multiple sensors [26]. In this paper we design an optimal distributed decision making algorithm robust to packet drops and implement it in SnowFort.

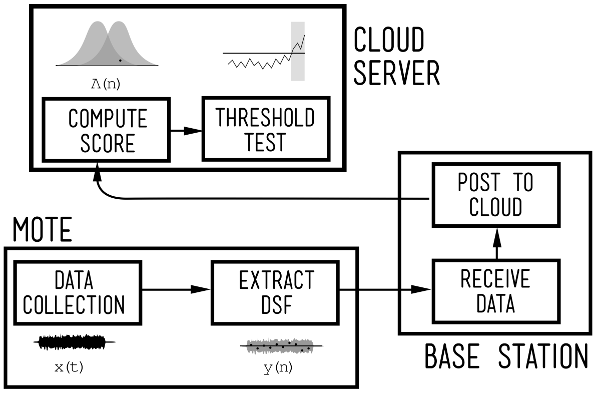

Fig. 8. Summary of the algorithm

The basic damage detection algorithm utilizes a Bayesian framework. The algorithm contains four steps as shown in Fig. 8: (1) Collect measurement; (2) Extract DSFs from measurements; (3) Compute scores ($\Lambda_n(Y)$) based on the Bayesian framework; (4) Detect damage by comparing the sequentially computed score and a threshold. The distributed implementation of the algorithm executes steps (1) and (2) in each mote, and then computes (3) and (4) in the cloud server. Computation of DSFs in the motes reduces the data transmission and computation of the decision rule on the cloud server maximizes the accuracy of the method. Each step of the method is detailed next.

The sampled data by each mote is divided into chunks of size N. An autoregressive model is fit to chunk r according to

$$x_r[n] = \sum_{k=1}^{p} a_k^r x_k[n-r] + \epsilon_r[n], \tag{1}$$

where a_k^r are the AR coefficients and $\epsilon_r[n]$ denotes the residual. The p values a_k^r for $k = 1, ..., p$ form a DSF vector and is transmitted by the mote to be stored in the cloud server. To simplify notation, we denote the n-th DSF a^n by $y[n]$.

The decision support system (DSS) in the cloud server computes a Bayesian decision in the following manner. Suppose damage happens at a random time λ. The DSF sequence $y[n]$ is identically and independently distributed with probability density function (pdf) f_0 before damage and pdf f_1 after damage. The test $P(\lambda \leq n|y[0], ..., y[n]) \geq 1-\alpha$ ensures that damage is detected with false alarm probability at most α. The test can be computed recursively as follows

$$\begin{aligned}
\Lambda_n(Y^n) &= \frac{P(\lambda \leq n|Y^n)}{P(\lambda > n|Y^n)} && (2)\\
&= \frac{\sum\limits_{k=0}^{n} \pi(k) \prod\limits_{r=1}^{k-1} f_0(y[r]) \prod\limits_{s=k}^{n} f_1(y[s])}{\sum\limits_{k=n+1}^{\infty} \pi(k) \prod\limits_{r=1}^{n} f_0(y[r])} && (3)\\
&= \frac{\sum\limits_{k=0}^{n} \pi(k) \prod\limits_{r=k}^{n} \frac{f_1(y[r])}{f_0(y[r])}}{\sum\limits_{k=n+1}^{\infty} \pi(k)} && (4)\\
&= \left(\frac{\Pi_{n-1}}{\Pi_n}\Lambda_{n-1} + \frac{\pi(n)}{\Pi_n}\right)\frac{f_1(y[n])}{f_0(y[n])}, && (5)
\end{aligned}$$

where $Y^n = \{y[t], 1 \leq t \leq n\}$, $\pi(k)$ denotes the prior distribution of λ (i.e., $\pi(n) = P(\lambda = n)$), and Π_n denotes the prior complementary cumulative distribution of λ (i.e., $\Pi_n = P(\lambda > n)$). A recursion form of the score computation is given in (5). When $n = 0$, $\Lambda_0 = \frac{\pi(0)}{1-\pi(0)}$. The damage detection test then reduces to $\Lambda_n(Y^n) \geq (1-\alpha)/\alpha$. The algorithm generalizes to multiple motes transmitting DSFs by aggregating them into a vector $\mathbf{y}[n]$ at time n. The corresponding densities are then defined as joint densities over all DSFs.

In WSN, packets may be lost or experienced interference during transmission. Therefore, some DSFs may not be received. When the score is computed, the time steps associated with the missing DSFs are disregarded. Let $[a, b]$ denote the integer set $\{x|x \in \mathbb{N}, a \leq x \leq b\}$, i.e. $[a, b] := \{a, a+1, a+2, \ldots, b-1, b\}$, $\mathcal{S}$ denote the set of the time steps when DSFs are missing, and $[a, b] \setminus \mathcal{S}$ denote a set contains all the integers in the set $[a, b]$ but not in $\mathcal{S}$, i.e. $\{x \in [a, b]\,|x \notin \mathcal{S}\}$. For example, if time step 3 and 6 are missing, then $\mathcal{S} = \{3, 6\}$. Therefore, the set $[2, 7] \setminus \mathcal{S}$ is $\{2, 4, 5, 7\}$. The score computation equation (4) becomes to

$$\Lambda_n(Y^n) = \frac{\sum\limits_{k\in[0,n]\setminus\mathcal{S}} \pi(k) \prod\limits_{r\in[k,n]\setminus\mathcal{S}} \frac{f_1(y[r])}{f_0(y[r])}}{\sum\limits_{k=n+1}^{\infty} \pi(k)}. \tag{6}$$

In case multiple sensors are utilized, the same effect is obtained by integrating the joint distributions $f_0(\mathbf{y})$ and $f_1(\mathbf{y})$ along the dimensions of missing samples.

VI. Experiments and Results

We evaluate the performance of SnowFort in two different ways. First, we measure the system performance with respect to communications, power consumption and synchornization performacne in particular. Second, we measure the performance of the system for damage detection in SHM we proposed.

A. System Performance

1) Setup: The network performance, power consumption and synchronization performance of SnowFort are evaluated on both an emulator and a physical hardware platform. Telosb motes are used in both tests.

For emulating a WSN, we use the COOJA emulator, which emulatues the hardware device and runs the firmware program with real device performance [48]. COOJA supports different hardware platforms with factory configurations and parameters and various realistic wireless communication mediums. It allows rapidly prototyping changes to the code of a large scale emulated network. As dissuced in [27], an emulation platform can reduce the development time of WSN. The emulation can be utilized by other SnowFort users as well. We set up a network with 16 Telosb motes, as shown in Fig. 9. Mote 1 serves as a base station. The other 15 motes are used for data collection and distributed in an area of 150 × 130 square meters. The emulation and experiment configuration is summarized in Table. II. The power profile tool Powertrace [49] is used to profile the power consumption of both the base station and data collection motes. The packet sequence number is used to trace missing packets. When the base station detects the discontinuity of packet sequence numbers, a missing packet is declared. All raw data are transmitted without compression. The size of the data frame payload is maximized. This setup helps us to explore a scenario with maximum power consumption.

TABLE II. System performance emulation configuration

Number of time slots	16
Time slot length	6.25 ms
Wireless medium	multi-path channel
Transmission power	0 dBm

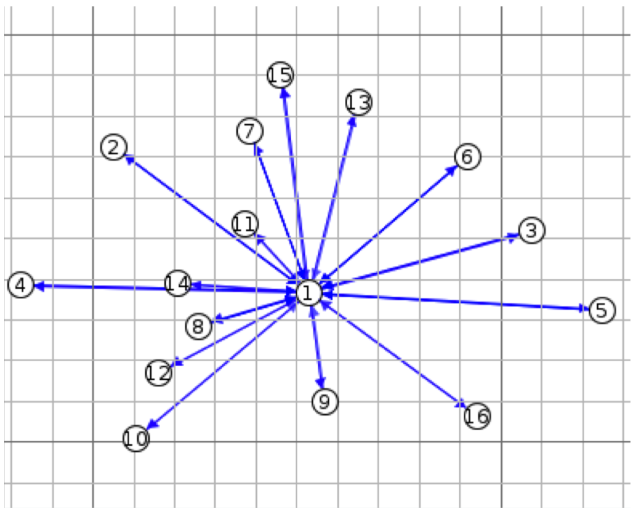

Fig. 9. Emulation Topology. Mote 1 is a base station.

For the real world experiment, the system setup and parameters are as same as the emulation. In addition, each Telosb mote is connected with a MPU 6050 I^{2}C sensor, which has one three-axis accelerometer, one three-axis gyroscope, and one temperature sensor. The I^{2}C sensor has an embedded anti-aliasing filter with configurable cut-off frequencies. The mote is powered by two Alkaline AA batteries with the supply voltage of 3 V. The I^{2}C sensor is powered by the mote via pin V_{cc} with the supply voltage of 2.7 V. Some pinout modifications are necessary to support the sensors.

TABLE III. Current consumption of Telosb mote [50] and MPU 6050 I^{2}C sensor [30].

Operation State	Current Consumption (normal)	Current Consumption (max)
CPU active	1.8 mA	2.4 mA
CPU inactive	5.1 μA	21 μA
radio TX	19.5 mA	21 mA
radio RX	21.8 mA	23 mA
gyroscope	3.6 mA	—
accelerometer	500 μA	—
gyroscope and accelerometer	3.8 mA	—

2) Power Consumption: For Telosb mote, there are four operation states: CPU active, CPU inactive, radio TX, and radio RX [50]. The power consumption of each status is summarized in Table III. (7) can be used to evaluate power consumption of the mote,

$$\begin{aligned} E_{\text{total}} &= E_{\text{active}} + E_{\text{inactive}} + E_{\text{TX}} + E_{\text{RX}} + E_{\text{Sensor}} \\ &= V_{\text{supply}} \times (I_{\text{active}} T_{\text{active}} + \\ &\quad I_{\text{inactive}} T_{\text{inactive}} + I_{\text{TX}} T_{\text{TX}} + \end{aligned}$$

$$
\begin{aligned}
& I_{\mathrm{RX}}T_{\mathrm{RX}}) + V_{\mathrm{Sensor}}I_{\mathrm{Sensor}}T_{\mathrm{Sensor}} && (7)\\
= \ & T_{\mathrm{total}}P_{\mathrm{total}} && (8)\\
= \ & T_{\mathrm{total}} \times (P_{\mathrm{active}} + P_{\mathrm{inactive}} + \\
& P_{\mathrm{TX}} + P_{\mathrm{RX}} + P_{\mathrm{Sensor}}), && (9)
\end{aligned}
$$

where $E_{\mathcal{S}}$, $P_{\mathcal{S}}$, and $I_{\mathcal{S}}$ denote the energy consumption, the power, and the current consumption, in the operation status $\mathcal{S}$ respectively, V_{supply} denotes the supply voltage, which typically is 3 volts, V_{Sensor} denotes the supply voltage of sensor via the pin of mote, and $T_{\mathcal{S}}$ denotes the time staying in the operation status $\mathcal{S}$. The operation status $\mathcal{S}$ includes active status, inactive status, transmission (TX) status, receipt (RX) status and sensor. In addition, $T_{\mathrm{total}} = T_{\mathrm{active}} + T_{\mathrm{inactive}}$. The power consumption of both I^{2}C and ADC sensors are represented as E_{Sensor} in (7). For sensors, the operation time T_{Sensor} is T_{total}.

Fig. 10 shows measurements of power consumption of both the base station and the mote in COOJA emulation and the experiment with physical hardware devices. The power of each operation status is shown as well. All experiments are conducted for 10 minutes and measurements of power consumption data are collected every one second. We calculate the total power consumption over 10 minutes and then compute P_{total} by using (8).

In the emulation, for the base station, the time consumed in active status, inactive status, TX status, and RX status is 63.4 s, 556.6 s 1.09 s, and 618.91 s respectively. According to (7) and Table. III, the total power consumption of the base station wireless transceiver is 40.891 jours. Since the total time length for measurement is 620 seconds, according to (8), the power is 65.94 mW. For motes, the average time consumed of each mote in each status is 47.306 s, 572.694 s, 2.133 s, and 6.652 s respectively. The average total power consumption of each mote is 824.013 mJ and the average power of each mote is 1.33 mW. The base station is typically setup with access to a regular power supply such as a solar panel or outlet. A standard Alkaline AA battery is usually 3.9 Wh, which is equivalent to 14040 J. Hence, with two AA batteries, the mote in SnowFort can operate for 244 days without battery replacement. The transmission power could be set to less than the maximum to extend lifetime further.

In the experiment with physical hardware devices, the power of the base station wireless transceiver is 65.78 mW and the mote without any sensors connected consumes 1.24 mW. When the mote is set to utilize the accelerometer alone it consumes 2.59 mW (lifetime 125 days) and when utilizing both accelerometer and gyro consumption is 11.50 mW (lifetime 28 days).

SnowFort saves power consumption significantly compared with other existing systems. In [51], the author proposes a WSN to monitor the ambient vibration of the Golden Gate Bridge. The sensor used in this monitoring system is powered by three high capacity lithium batteries. The estimated lifetime is 23 days. By using the same battery type, the SnowFort system can operate over 3 years. In [52], the WISAN node also has a MSP430 microprocessor and a CC2420 radio transceiver but has no sensor. It has a peak power consumption of 75 mW, which is larger than the power consumption of our system by a factor of 60. Several power saving optimizations are introduced in [52].

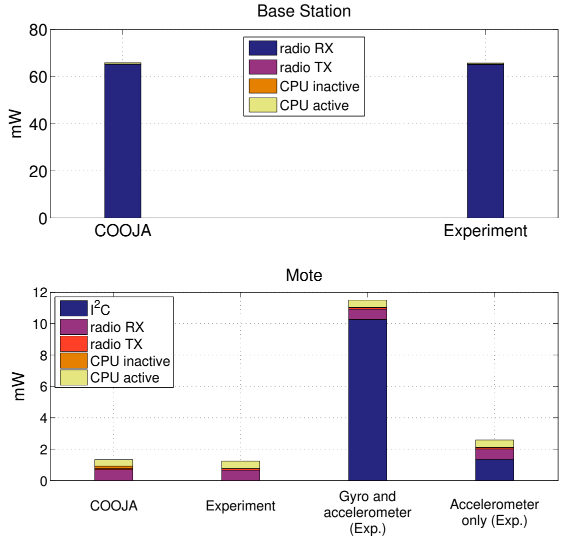

Fig. 10. Operating power of mote. The upper plot shows the power of the base station. The lower plot shows the power of the mote.

A recently proposed wireless sensing unit with an accelerometer has a power consumption of 1.73 mW [33]. The custom accelerometer in their test has power consumption of 0.35 mW. If this accelerometer was used in place of the MPU 6050 sensor, the mote in SnowFort would have total consumption of 1.59 mW. The test in [33] is done with a single sensor, but since their system uses standard CSMA/CA, significantly higher power consumption can be expected.

3) Network Performance: We use (10) to compute the packet drop rate (PDR),

$$
\mathrm{PDR} = \left(1 - \frac{\text{Number of packets received}}{\text{Number of packets transmitted}}\right) \times 100\%. \tag{10}
$$

For the COOJA emulation, the PDR is 0.012%. For the real-world experiment, the PDR is 0.959%. In [34], authors test a network with 10 motes utilizing a standard CSMA/CA protocol with IEEE 802.15.4 radios. The average reported packet delay is 55 ms. For each packet, the expected transmission time is 8.7 ms. The PDR for the CSMA/CA scheme is approximately 86.34%.

The SnowFort system can support up to 54 motes at 32Hz in the experimental configuration. In this configuration, the PDR is 0.0154% in COOJA emulation. The limitation is the processing time for each packet required at the base station. Additional low cost base stations can be added to support additional motes.

4) Sychronization Performance: The time synchronization accuracy is tested by sampling a sinusoid signal on four motes associated with two different base stations. The time delay is computed from the differences between the phase offset estimates of any two motes, i.e.

$$\Delta_{ij} = \frac{|\phi_i - \phi_j|}{2\pi f_0}, \tag{11}$$

where f_0 denotes the frequency of the sinusoid signal. By repeating the experiments multiple times, the average time delay is 98μs, which meets the 1ms minimum requirement of SHM, as suggested by [53]. For applications require higher accuracy, the present synchronization algorithm can be modified to the ones in [54] and [55].

B. Structural Health Monitoring Experiment

1) Setup: Data collected from a shake table experiment, which is conducted at the State University of New York at Buffalo [26], [56], are used to verify the performance of both SnowFort and the SHM damage detection algorithm. The acceleration data were recorded from each floor and the roof of a four-story frame. 5000 samples of data were recorded for earthquakes of three intensity levels: service level earthquake (SLE), design level earthquake (DLE) and collapse level earthquake (CLE). The samples collected during SLE and DLE are regarded as pre-damage state and at CLE as post-damage.

The data from the experiment is utilized to seed each mote to emulate the data collection and decision process. In the experiment, five Telosb motes are used to collect acceleration data and communicate with a common base station. The time slot interval is 0.8s. Signals collected by ADCs of motes are continuous time signals that are reconstructed by the discrete time data collected in the original shake table experiment. The analog signals are filtered by a analog anti-aliasing filter before sampling. Therefore, one mote virtually collects data from one floor or the roof of the 4-story structure. The sampling frequency of ADCs remain the same, which is $f_s = 128$ Hz. Motes fit a 3rd order AR model with chunked data. The chunk size is set to $N = 100$. The DSF is the first AR coefficient. During each time slot, 5 DSFs are sent. Data decompression is performed at the base station after receiving DSFs. DSFs are then forwarded to the cloud server via the Internet. After receiving DSFs, the cloud server computes the score and performs threshold tests, as shown in Fig. 8.

2) Data Compression Performance: The data compression ratio (CR) is computed as in (12).

$$\begin{aligned} \text{CR} \quad &= \left(1 - \frac{\text{Total number of bits transmitted}}{\text{Total number of bits sampled}}\right) \\ &\quad \times 100\%. \end{aligned} \tag{12}$$

Fig. 11 shows CRs of each floor and the roof (5th floor). The overall average compression ratio is 68.55%. The compression system directly encodes the DSF which is highly correlated and follows an approximately Gaussian distribution [25], [26]. Therefore, there is no need to compute DSF residuals before compression.

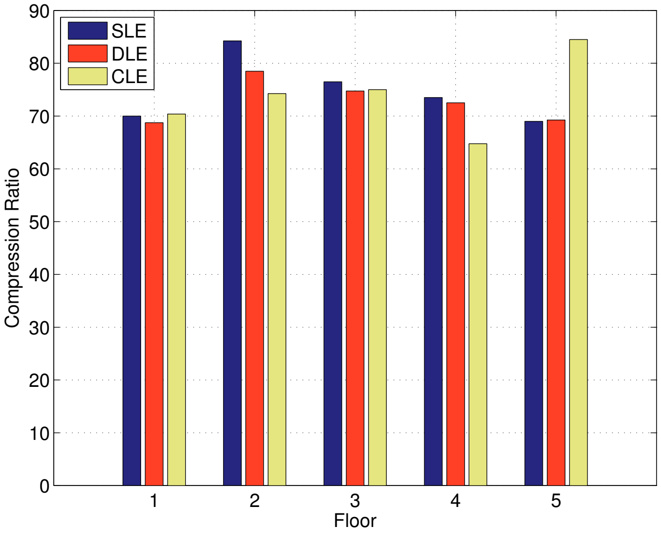

Fig. 11. Data compression performance on DSFs

3) Detection Algorithm Performance: In the validation experiment a total of 50 DSFs are generated for each type of earthquake signal on each floor. The pre-damage and post-damage distributions are assumed to be Gaussian distributions. The means and variances of f_0 and f_1 are computed based on pre-damage and post-damage DSFs respectively. The probability of false alarm α is set to 10^{-6}. When the score Λ_n is larger than the threshold $\tau = (1 - \alpha)/\alpha$, a damage is declared.

The upper plot of Fig. 12 shows scores of each floor and the roof. The horizontal dash line is the threshold. When the score surpasses the dash line, a damage is declared. In this experiment, the damage occurs on the roof and the true damage starts at $n = 101$. The sensors on the 2nd floor, 3rd floor, 4th floor, and the roof detect the roof damage at $n = 110$, $n = 105$, $n = 108$, and $n = 103$, respectively. The results are consistent with those presented for the centralized and complete data algorithm in [26]. The SnowFort monitoring system works as expected.

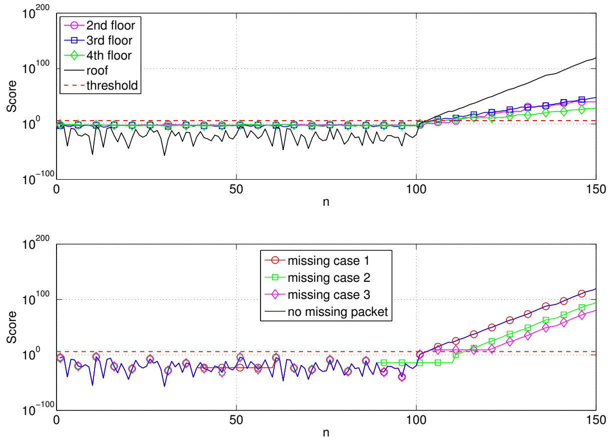

Fig. 12. Score Λ_n v.s. the time step n for $\alpha = 10^{-6}$.The upper plot shows scores without packet loss. The lower plot shows scores with packet loss.

The lower plot of Fig. 12 shows scores of the algorithm with packet loss. The acceleration signal from the roof is used as the testing signal. For the first test case, time steps from 40 to 60 are dropped. It is a pre-damage case. For the second test case, time steps from 90 to 110 are dropped. It is a damage transition case. The third test case is the post-damage case. DSF with time steps from 105 to 120 are missing. True damage starts at $n = 101$ analogous to the prior setup. In the benchmark scenario without packet loss, the fault is detected at $n = 103$. For the first case and the third case, faults are also detected at $n = 103$. For the second case, the fault is detected at $n = 114$, which is mostly introduced by missing DSFs. The detection delay caused by the detection algorithm is 3, which only needs one more DSF than the benchmark signal.

Fig. 13 demonstrates the performance of the detection algorithm with packet loss. The signal collected from the roof is used here. Expected detection delays are computed over 1000 realizations with PDR from 0.1% to 50%. As shown in Fig. 13, when PDR is small, the expected detection delay is 2. When PDR grows to 50%, the algorithm can still detect the damage with an expected delay of 3.8 time steps. Other floors have similar performances.

VII. Summary and Discussion

In this paper, SnowFort, a new WSN for infrastructure and environmental monitoring, is presented. This system introduces a new architecture for the integration of both a WSN and a decision support system, with real-time visualization, analytics, and interaction over a web interface. The optimization of the communication scheme, TDMA, not only improves the reliability and scalability of the network but also extends the lifetime of motes: when powered just by two AA batteries, they can persist for 244 days without sensing and 125 days with accelerometers and temperature sensors. The power comsumption is reduced by minimizing radio on time in TDMA, transmitting the compressed data, and further decreased by sending the features rather than the raw data in SHM experiment. A standard and RESTful API for data posting and retrieval makes the data visualization and analytical tools of SnowFort available to other systems. When integrated into SHM systems, SnowFort offers a low-power and reliable platform for detecting damage through a quantitative and Bayesian framework. The work for data processing is embeddable into each sensor mote, allowing for more efficient paths of communication within a distributed damage detection framework. Minimizing the effect of packet loss is also explored, aided by a decision mechanism that incorporates uncertainty, which has been validated through experiments.

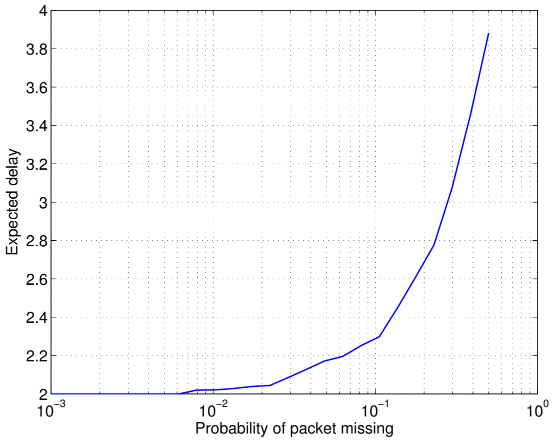

Fig. 13. Probability of packet missing v.s. expected detection delay

Acknowledgment

The authors would like to thank Prof. Anne Kiremidjian and Konstantinos Balafas for their comments and suggestions. This research was supported by NSF Grants and the Stanford TomKat Center for Sustainable Energy.

References

[1] J. P. Lynch, K. H. Law, A. S. Kiremidjian, T. Kenny, and E. Carryer, “A wireless modular monitoring system for civil structures,” in *Proc. the 20th International Modal Analysis Conference (IMAC XX)*, 2002, pp. 1–6.

[2] Y. Wang, J. P. Lynch, and K. H. Law, “A wireless structural health monitoring system with multithreaded sensing devices: design and validation,” *Structure and Infrastructure Engineering: Maintenance, Management, Life-Cycle Design and Performance*, vol. 3, no. 2, pp. 103–120, 2007.

[3] T. Nagayama and B. F. Spencer Jr, "Structural health monitoring using smart sensors," University of Illinois at Urbana-Champaign, Tech. Rep., 2007.

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

[5] 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 *Proc. ACM IPSN '09*. IEEE, 2009, pp. 277–288.

[6] A. Rowe, M. E. Berges, G. Bhatia, E. Goldman, R. Rajkumar, J. H. Garrett, J. M. Moura, and L. Soibelman, "Sensor andrew: Large-scale campus-wide sensing and actuation," *IBM Journal of Research and Development*, vol. 55, no. 1.2, pp. 6–1, 2011.

[7] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *Proc. ACM WSNA '02*. ACM, 2002, pp. 88–97.

[8] R. Szewczyk, E. Osterweil, J. Polastre, M. Hamilton, A. Mainwaring, and D. Estrin, "Habitat monitoring with sensor networks," *Communications of the ACM - Wireless Sensor Networks*, vol. 47, no. 6, pp. 34–40, Jun. 2004.

[9] 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 *Proc. ACM/IEEE IPSN '07*. ACM/IEEE, 2007, pp. 254–263.

[10] R. Bajwa, R. Rajagopal, P. Varaiya, and R. Kavaler, "In-pavement wireless sensor network for vehicle classification," in *Proc. IEEE IPSN '11*. IEEE, Apr. 2011, pp. 85–96.

[11] I. Stoianov, L. Nachman, S. Madden, T. Tokmouline, and M. Csail, "Pipenet: A wireless sensor network for pipeline monitoring," in *Proc. ACM/IEEE IPSN '07*. ACM/IEEE, 2007, pp. 264–273.

[12] G. Werner-Allen, K. Lorincz, M. Ruiz, O. Marcillo, J. Johnson, J. Lees, and M. Welsh, "Deploying a wireless sensor network on an active volcano," *IEEE Internet Comput.*, vol. 10, no. 2, pp. 18–25, 2006.

[13] G. Barrenetxea, F. Ingelrest, G. Schaefer, M. Vetterli, O. Couach, and M. Parlange, "Sensorscope: Out-of-the-box environmental monitoring," in *Proc. ACM/IEEE IPSN '08*. IEEE, 2008, pp. 332–343.

[14] D. Culler, J. Hill, M. Horton, K. Pister, R. Szewczyk, and A. Woo, "Mica: The commercialization of microsensor motes," *Sensors Magazine*, vol. 19, no. 4, pp. 40–48, 2002.

[15] Snowfort website. Accessed: September 27, 2014. [Online]. Available: http://snowfort.stanford.edu

[16] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "Tinydb: An acquisitional query processing system for sensor networks," *ACM Trans. Database Syst.*, vol. 30, no. 1, pp. 122–173, Mar. 2005.

[17] R. Mueller, G. Alonso, and D. Kossmann, "Swissqm: Next generation data processing in sensor networks," in *Proc. Innovative Data Systems Research 2007*, vol. 7, 2007, pp. 1–9.

[18] B. F. Spencer Jr and C.-B. Yun, "Wireless sensor advances and applications for civil infrastructure monitoring," Newmark Structural Engineering Laboratory. University of Illinois at Urbana-Champaign., Tech. Rep., 2010.

[19] H. Jo, S.-H. Sim, K. A. Mechitov, R. Kim, J. Li, P. Moinzadeh, B. F. Spencer, Jr., J. W. Park, S. Cho, H.-J. Jung, C.-B. Yun, J. A. Rice, and T. Nagayama, "Hybrid wireless smart sensor network for full-scale structural health monitoring of a cable-stayed bridge," in *Proc. SPIE Smart Structures/NDE Conference 2011*, vol. 7981, 2011, p. 15.

[20] J. A. Rice, K. Mechitov, S.-H. Sim, T. Nagayama, S. Jang, R. Kim, B. F. Spencer Jr, G. Agha, and Y. Fujino, "Flexible smart sensor framework for autonomous structural health monitoring," *Smart Structures and Systems*, vol. 6, no. 5-6, pp. 423–438, 2010.

[21] A. S. Kiremidjian, G. Kiremidjian, and P. Sarabandi, "A wireless structural monitoring system with embedded damage algorithms and decision support system," *Structure and Infrastructure Engineering: Maintenance, Management, Life-Cycle Design and Performance*, vol. 7, no. 12, pp. 881–894, Dec. 2011.

[22] G. Fortino, A. Guerrieri, G. M. O'Hare, and A. Ruzzelli, "A flexible building management framework based on wireless sensor and actuator networks," *Journal of Network and Computer Applications*, vol. 35, no. 6, pp. 1934–1952, 2012.

[23] C. de Farias, H. Soares, L. Pirmez, F. Delicato, I. Santos, L. F. Carmo, J. de Souza, A. Zomaya, and M. Dohler, "A control and decision system for smart buildings using wireless sensor and actuator networks," *Transactions on Emerging Telecommunications Technologies*, vol. 25, no. 1, pp. 120–135, 2014.

[24] B. Balaji, J. Xu, A. Nwokafor, R. Gupta, and Y. Agarwal, "Sentinel: occupancy based hvac actuation using existing wifi infrastructure within commercial buildings," in *Proc. ACM SenSys '13*. ACM, Nov. 2013, pp. 17:1–17:14.

[25] H. Y. Noh, K. K. Nair, A. S. Kiremidjian, and C. Loh, "Application of time series based damage detection algorithms to the benchmark experiment at the national center for research on earthquake engineering (ncree) in taipei, taiwan," *Smart Structures and Systems*, vol. 5, no. 1, pp. 95–117, 2009.

[26] H. Noh, R. Rajagopal, and A. Kiremidjian, "Sequential structural damage diagnosis algorithm using a change point detection method," *Journal of Sound and Vibration*, vol. 332, no. 24, pp. 6419–6433, Nov. 2013.

[27] G. Hackmann, W. Guo, G. Yan, Z. Sun, C. Lu, and S. Dyke, "Cyber-physical codesign of distributed structural health monitoring with wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 1, pp. 63–72, 2014.

[28] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki-a lightweight and flexible operating system for tiny networked sensors," in *Proc. IEEE 29th Annual International Conference on Local Computer Networks, 2004*. IEEE, Nov. 2004, pp. 455–462.

[29] J. Polastre, R. Szewczyk, and D. Culler, "Telos: enabling ultra-low power wireless research," in *Proc. IEEE IPSN '05*. IEEE, 2005, pp. 364–369.

[30] I. Inc. Mpu-6000 and mpu-6050 product specification revision 3.4. Accessed: September 27, 2014. [Online]. Available: http://www.invensense.com/mems/gyro/documents/PS-MPU-6000A-00v3.4.pdf

[31] C. Wu, Y. Xu, Y. Chen, and C. Lu, "Submodular game for distributed application allocation in shared sensor networks," in *Proc. IEEE INFOCOM '12*. IEEE, March 2012, pp. 127–135.

[32] Y. Fu, M. Sha, C. Wu, A. Kutta, A. Leavey, C. Lu, H. Gonzalez, W. Wang, B. Drake, Y. Chen, and P. Biswas, "Thermal modeling for a hvac controlled real-life auditorium," in *International Conference on Distributed Computing Systems*, Jul. 2014.

[33] T. Torfs, T. Sterken, S. Brebels, J. Santana, R. van den Hoven, V. Spiering, N. Bertsch, D. Trapani, and D. Zonta, "Low power wireless sensor network for building monitoring," *IEEE Sensors J.*, vol. 13, no. 3, pp. 909–915, 2013.

[34] W.-T. Huang, J.-T. Lin, C.-H. Chen, Y.-J. Chang, and Y.-Y. Chen, "Accurate analysis of ieee 802.15. 4 slotted csma/ca over a real-time wireless sensor network," in *Sensor Systems and Software*. Springer, 2010, pp. 98–110.

[35] A. El-Hoiydi and J.-D. Decotignie, "Wisemac: an ultra low power mac protocol for the downlink of infrastructure wireless sensor networks," in *Proc. IEEE ISCC '04*, vol. 1. IEEE, Jun. 2004, pp. 244–251.

[36] A. Dunkels, "The contikimac radio duty cycling protocol," Swedish Institute of Computer Science, Tech. Rep., Mar. 2011.

[37] M. Buettner, G. V. Yee, E. Anderson, and R. Han, "X-mac: a short preamble mac protocol for duty-cycled wireless sensor networks," in *Proc. ACM SenSys '06*. ACM, 2006, pp. 307–320.

[38] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proc. IEEE 33rd Annual Hawaii International Conference on System Sciences*. IEEE, 2000, p. 10.

[39] I. Demirkol, C. Ersoy, and F. Alagoz, "Mac protocols for wireless sensor networks: a survey," *IEEE Commun. Mag.*, vol. 44, no. 4, pp. 115–121, Apr. 2006.

[40] P. Huang, L. Xiao, S. Soltani, M. W. Mutka, and N. Xi, "The evolution of mac protocols in wireless sensor networks: A survey," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 1, pp. 101–120, Feb. 2013.

[41] S. C. Ergen and P. Varaiya, "Tdma scheduling algorithms for wireless sensor networks," *Wireless Networks*, vol. 16, no. 4, pp. 985–997, 2010.

[42] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient mac protocol for wireless sensor networks," in *Proc. IEEE INFOCOM '02.*, vol. 3. IEEE, Jun. 2002, pp. 1567–1576.

[43] T. Van Dam and K. Langendoen, "An adaptive energy-efficient mac protocol for wireless sensor networks," in *Proc. ACM SenSys '03*. ACM, 2003, pp. 171–180.

[44] S. Gobriel, D. Mosse, and R. Cleric, "Tdma-asap: Sensor network tdma scheduling with adaptive slot-stealing and parallelism," in *Proc. IEEE ICDCS '09*. IEEE, 2009, pp. 458–465.

[45] A. Goldsmith, *Wireless communications*. Cambridge university press, 2005.

[46] F. Marcelloni and M. Vecchio, "A simple algorithm for data compression in wireless sensor networks," *IEEE Commun. Lett.*, vol. 12, no. 6, pp. 411–413, Jun. 2008.

[47] V. A. Kottapalli, A. S. Kiremidjian, J. P. Lynch, E. Carryer, T. W. Kenny, K. H. Law, and Y. Lei, "Two-tiered wireless sensor network architecture for structural health monitoring," in *Proc. SPIE Smart Structures and Materials 2003: Smart Systems and Nondestructive Evaluation for Civil Infrastructures*. SPIE, 2003, pp. 8–19.

[48] J. Eriksson, F. Österlind, N. Finne, N. Tsiftes, A. Dunkels, T. Voigt, R. Sauter, and P. J. Marrón, "Cooja/mspsim: interoperability testing for wireless sensor networks," in *Proc. Simutools '09*. ICST, 2009, p. 27.

[49] A. Dunkels, J. Eriksson, N. Finne, and N. Tsiftes, "Powertrace: Network-level power profiling for low-power wireless networks," Swedish Institute of Computer Science, Tech. Rep., Mar. 2011.

[50] M. Corporation. Tmote sky datasheet. Accessed: September 27, 2014. [Online]. Available: http://www.eecs.harvard.edu/~konrad/projects/shimmer/references/tmote-sky-datasheet.pdf

[51] S. Kim, "Wireless sensor networks for structural health monitoring," Master's thesis, UC BERKELEY, 2005.

[52] E. Sazonov, K. Janoyan, and R. Jha, "Wireless intelligent sensor network for autonomous structural health monitoring," in *Proc. SPIE Smart Structures and Materials 2004: Smart Sensor Technology and Measurement Systems*, vol. 5384. International Society for Optics and Photonics, Mar. 2004, pp. 305–314.

[53] F. Pentaris, J. Stonham, and J. Makris, "A review of the state-of-the-art of wireless shm systems and an experimental set-up towards an improved design," in *Proc. IEEE EUROCON'13*. IEEE, 2013, pp. 275–282.

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

[55] E. Sazonov, V. Krishnamurthy, and R. Schilling, "Wireless intelligent sensor and actuator network-a scalable platform for time-synchronous applications of structural health monitoring," *Structural Health Monitoring*, vol. 9, no. 5, pp. 465–476, 2010.

[56] D. Lignos, H. Krawinkler, and A. Whittaker, "Shaking table collapse tests of a 4–story steel moment frame," in *Proc. 14th World Conference in Earthquake Engineering*, 2008, pp. 12–17.

Yizheng Liao (S08) received the M.Sc. and B.Sc. degrees in electrical and computer engineering from Worcester Polytechnic Institute, Worcester, MA, in 2011. He received the M.S. in electrical engineering from Stanford University, Stanford, CA, in 2013. He is currently working toward the Ph.D. degree in the Department of Civil and Environmental Engineering at Stanford University. His research interests include wireless sensor network, structural health monitoring, and change point detection.

Mark Mollineaux is a PhD candidate at Stanford University, specializing in SHM applications with novel sensors.

Richard Hsu received the B.S. degree in electrical engineering and computer science from the University of California, Berkeley and the M.S. degree in computer science at Stanford University.

Rebekah Bartlett received the B.S.E. degree in electrical engineering and B.S. degrees in Mathematical Sciences and Interdisciplinary Physics in 2012 from the University of Michigan at Ann Arbor. After graduation she spent eight months with the Advanced Sensor Techniques group at MIT Lincoln Laboratory working on distributed beamforming before starting graduate school. Her current research interests include wireless networks and embedded systems.

Ravneet Bajwa received his Phd (2013), MS (2010), and BS (2007) in Electrical Engineering and Computer Sciences from UC Berkeley. His research work focused on using road vibrations, measured by ultra-low power wireless sensors, to automatically detect, classify, and weigh on-going vehicles. His current research interests include wireless sensor design, signal processing, and machine learning. He is very passionate about building low-power smart devices that can be connected to the internet.

Anubhav Singla received the B.Tech degree in Electrical Engineering from Indian Institute of Technology, Kanpur, India in 2011 and M.S. degree in Electrical Engineering degree from Stanford University, Stanford, CA, in 2013. Since 2012 he has been with Qualcomm Technologies Inc., Santa Clara, CA. His research interests include applications of wireless communication, signal processing and optimization.

Adnan Baja obtained his M.S and Ph.D. in Electrical and Computer Engineering from the University of Illinois, Urbana-Champaign in 2008 and 2012 respectively and B.Tech in Electrical Engineering from IIT Bombay in 2006. Adnan Raja is currently working as a Systems Engineer at Fastback Networks.

Prior to joining Fastback, Adnan was a post-doctoral researcher at Stanford University where he led a project in designing a large-scale low-power wireless sensor network for infrastructure monitoring.

Adnan has also interned at Qualcomm Inc. on various wireless technologies like 4G, White space (cognitive radio) communication, Peer-to-peer communication, Interference cancellation schemes in cellular networks.

Ram Rajagopal received the bachelors degree in electrical engineering from the Federal University of Rio de Janeiro, Brazil, the masters degree in electrical and computer engineering from University of Texas, Austin, TX, USA, and the M.A. degree in statistics and the Ph.D. degree in electrical engineering and computer sciences from the University of California, Berkeley, CA, USA.

He is an Assistant Professor of Civil and Environmental Engineering at Stanford University, Stanford, CA, USA, where he directs the Stanford Sustainable Systems Lab (S3L), focused on large scale monitoring, data analytics and stochastic control for infrastructure networks, in particular energy and transportation. His current research interests in power systems are in integration of renewables, smart distribution systems and demand-side data analytics. Prior to his current position he was a DSP Research Engineer at National Instruments and a Visiting Research Scientist at IBM Research. He holds more than 30 patents from his work, and has advised or founded various companies in the fields of sensor networks, power systems and data analytics.

Dr. Rajagopal is a recipient of the Powell Foundation Fellowship, Berkeley Regents Fellowship and the Makhoul Conjecture Challenge award.