e-Sampling: Event-Sensitive Autonomous Adaptive Sensing and Low-Cost Monitoring in Networked Sensing Systems

MD ZAKIRUL ALAM BHUIYAN, Guangzhou University and Fordham University
JIE WU, Temple University
GUOJUN WANG, Guangzhou University
TIAN WANG, Huaqiao University
MOHAMMAD MEHEDI HASSAN, King Saud University

Sampling rate adaptation is a critical issue in many resource-constrained networked systems, including Wireless Sensor Networks (WSNs). Existing algorithms are primarily employed to detect events such as objects or physical changes at a high, low, or fixed frequency sampling usually adapted by a central unit or a sink, therefore requiring additional resource usage. Additionally, this algorithm potentially makes a network unable to capture a dynamic change or event of interest, which therefore affects monitoring quality. This article studies the problem of a fully autonomous adaptive sampling regarding the presence of a change or event. We propose a novel scheme, termed "event-sensitive adaptive `sampling` and low-cost monitoring (`e-Sampling`)" by addressing the problem in two stages, which leads to reduced resource usage (e.g., energy, radio bandwidth). First, `e-Sampling` provides the embedded algorithm to adaptive sampling that automatically switches between high- and low-frequency intervals to reduce the resource usage, while minimizing false negative detections. Second, by analyzing the frequency content, `e-Sampling` presents an event identification algorithm suitable for decentralized computing in resource-constrained networks. In the absence of an event, the "uninteresting" data is not transmitted to the sink. Thus, the energy cost is further reduced. `e-Sampling` can be useful in a broad range of applications. We apply `e-Sampling` to Structural Health Monitoring (SHM) and Fire Event Monitoring (FEM), which are typical applications of high-frequency events. Evaluation via both simulations and experiments validates the advantages of `e-Sampling` in low-cost event monitoring, and in effectively expanding the capacity of WSNs for high data rate applications.

Categories and Subject Descriptors: C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design; C.2.4 [**Computer-Communication Networks**]: Distributed Systems; C.4 [**Computer-Communication Networks**]: Performance of Systems; H.1.1 [**Systems and Information**]: Value of information

A preliminary version of this work has appeared [Bhuiyan et al. 2013] in *Proceedings of the 10th IEEE International Conference on Sensing, Communication, and Networking (SECON'13)*.

This research is supported in part by NSF Grants No. CNS 1449860, No. CNS 1461932, No. CNS 1460971, No. CNS 1439672, No. CNS 1301774, and No. ECCS 1231461; in part by the National Natural Science Foundation of China under Grants No. 61632009, No. 61402543, and No. 61472451, and High Level Talents Program of Higher Education in Guangdong Province under Funding Support Number 2016ZJ01. The authors would also like to extend their sincere appreciation to the Deanship of Scientific Research at King Saud University for its funding of this research through the Research Group Project No. RGP-281.

Authors' addresses: M. Z. A. Bhuiyan, School of Computer Science and Educational Software, Guangzhou University, Guangzhou 510006, China, and Department of Computer and Information Sciences, Fordham University, New York, NY, 10458; email: mbhuiyan3@fordham.edu; J. Wu, Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122; email: jiewu@temple.edu; G. Wang (corresponding author), School of Computer Science and Educational Software, Guangzhou University, Guangzhou 510006, China; email: csgjwang@gmail.com; T. Wang, College of Computer Science and Technology, Huaqiao University, Xiamen, China, 361021; email: wangtian@hqu.edu.cn; M. M. Hassan, King Saud University, Saudi Arabia 12372; email: mmhassan@ksu.edu.sa.

DOI: http://dx.doi.org/10.1145/2994150

General Terms: Design, Algorithms, Measurement, Performance

Additional Key Words and Phrases: Wireless sensor networks, decentralized signal processing, adaptive sampling, decentralized decision making, event monitoring, energy-efficiency

ACM Reference Format:
Md Zakirul Alam Bhuiyan, Jie Wu, Guojun Wang, Tian Wang, and Mohammad Mehedi Hassan. 2017. e-Sampling: Event-sensitive autonomous adaptive sensing and low-cost monitoring in networked sensing systems. ACM Trans. Auton. Adapt. Syst. 12, 1, Article 1 (March 2017), 29 pages.
DOI: http://dx.doi.org/10.1145/2994150

1. INTRODUCTION

In recent years, there has been a remarkable growth and momentum of large-scale networked sensing systems based on sensor networks and mobile devices, with increasing sophistication and diversity of sensors [Xu et al. 2015; Aderohunmu et al. 2015; Khan et al. 2013; Alam et al. 2015; Bhuiyan et al. 2015a]. Such systems support a lot of diverse event monitoring applications, including traffic, road condition, irregular noise, air quality, pollution, Structural Health Monitoring (SHM), physical activity monitoring, and environmental monitoring (e.g., wildlife, fire, snow). Sensor nodes in these systems, such as Wireless Sensor Networks (WSNs) are expected to work autonomously to support long-lived and inexpensive acquisition of data from the physical world. Therefore, the problem of low-power consumption of nodes is an important issue. Due to the network growth in scale and severe resource constraints—in particular, energy, computation, and bandwidth—a sound body of literature has centered on extending the network lifetime from different perspectives—adaptive sampling, routing efficiency, period monitoring, etc.

On the one hand, the networks are expected to monitor events in those applications on a long-term basis. However, sensors generate too much data for their radios in these applications, especially in those that involve audio, seismometers, imaging, and vibration [Luo et al. 2010; Chen et al. 2012; Bhuiyan et al. 2015b, 2015c; Wang et al. 2010a]. In most cases, they cannot send that data in even one hop in real-time, due to limited bandwidth. Frequent transmission of such raw data, even a reduced amount of the data, results in significant data loss, due to channel contention (presuming a multiple access MAC) and network congestion, especially in the case of medium to large-scale networks. All of these applications bring challenges to sensors' resource circumstances. Thus, sensors should be able to reduce data autonomously before transmission.

Sampling is the signal measurement at regular intervals obtained by the reduction of a continuous signal to a discrete-time signal (a sequence of samples). A *sample* is a value or set of values at a point in time. The *sampling frequency*, or *sampling rate* is then the rate at which new samples are taken from the continuous signal. An *optimal sampling rate* at a point in time can be defined by either the Nyquist rate or a maximal sampling rate corresponding to frequency content. The idea of sampling rate adaptation in extending the lifetime of resource-constrained networks is not new and adaptive sampling methods/algorithms are exploited to reduce resource usage in networks [Xu et al. 2015; Chen et al. 2012; Hao et al. 2015; Aderohunmu et al. 2015; Chatterjea and Havinga 2009; Bhuiyan et al. 2015a; Hao et al. 2015]. State-of-the-art schemes to adaptive sampling focus on different concerns, like aggressively reducing the spatial sampling rate of sensor, assigning the optimal sampling rate, utility-based sensing and communication, sample scheduling, and compressed sensing [Wang et al. 2010b; Kho et al. 2009; Shu et al. 2008; Luo et al. 2010; Alippi et al. 2010; Willet et al. 2004; Chen et al. 2012; Masoum et al. 2012]. Some of the concerns are interrelated and considered in conjunction with each other. All of these schemes aim to reduce the sensor energy cost through the reduction of hardware activity and data acquisition and communication activity, and they advance the networks' operational performance for various types of event detection. However, there still exists a set of crucial concerns in high-frequency physical event detection, particularly in cyberphysical system applications [Hackmann et al. 2014]. Some of these concerns are summarized as follows:

- Sensors cannot take actions autonomously to switch their rates and intervals. They all need to rely on an omniscient central unit or a sink that knows the signal complexities of all of the sensor positions and periodically transmits suitable sampling rates to them. Consequently, this adaptation induces extra communication overhead to traffic-sensitive networks.
- It is difficult to assign sampling rates or allocate bandwidth to sensors in a specific region where an interesting event occurs, especially in the case of "emergency" alarming applications, such as fire, damage to the structures.
- Due to unreliable wireless communications, the sink may obtain incomplete or sometimes suspicious information, which leads to inaccurate judgments on rate adaptation. This may make sensor nodes in a network unable to capture a dynamic change in the environment, which degrades the quality of monitoring.
- Waiting for messages from the sink about an appropriate sampling rate and bandwidth allocation may lead to a dynamic event undetected in some applications such as wildlife monitoring, unauthorized intrusion. This means that that the event may disappear by the time the sensors get their rates from the sink.

On the other hand, natural environments are often extremely dynamic, where the presence of a physical event in many applications is also dynamic. Some events may not appear once in hours, days, months, even years (e.g., damage, fire, snow level, flood, and so on [Alippi et al. 2010]). Thus, sensors are required to continuously adjust their activities to dynamic systems (e.g., cyberphysical systems [Hackmann et al. 2014; Bhuiyan et al. 2015b, 2016]). The challenge is to represent an accurate picture of changes in the event process and environmental variables. This can only be achieved if the event is sensed or sampled from the environment at an accurate rate. Thus, sampling rate should be regarded as a function of both the dynamic physical phenomena and the application-specific.

Motivated by the limitations and requirements, this article studies the problem of autonomous adaptive sampling regarding the presence of a change/event. We propose a novel scheme, which we have termed "event-sensitive adaptive `sampling` and low-cost monitoring (`e-Sampling`)," by addressing the problem in two stages, which lead to reduced resource usage such as energy and radio bandwidth in networks, while maintaining high accuracy of monitoring. First, `e-Sampling` provides the embedded algorithm to adaptive sampling that automatically switches between high- and low-frequency intervals to reduce the resource usage while minimizing false negative detections. Particularly, each sensor has "short" and recurrent "bursts" of high-rate sampling, and samples at a much lower rate at any other time. Depending on the analysis of the frequency content of signals, whenever one of the short intervals of high-rate sampling is longer than normal, possibly due to the presence of an interesting event, the observations (the frequency content of signals) become important. By having an embedded algorithm, each sensor autonomously and automatically switches (takes actions on) its rates and both the high- and low-rate intervals. Previously discussed limitations are overcome to a great extent because `e-Sampling` enables reliable analysis to estimate appropriate future sampling rates and net reduction in acquired samples.

In the second stage, `e-Sampling` enables sensors to compute a lightweight indication of the presence of an event by analyzing only the important frequency content in a decentralized manner. A significant change in the content, which is called *event-sensitive* or *interesting data*, indicates that a possible event occurred in a given monitoring application. If the event has truly occurred, the sink who receives the indications may want detailed information from the sensors in specific regions (e.g., which are located around the event) and may ask queries; otherwise, in the absence of the event (if the strength of the event indication is less than expected), sensors reduce data (called *uninteresting data*) transmission to the sink. This scheme is quite flexible—it supports several very different applications (described previously) and is completely autonomous. In addition, it can be applied to spectrum sensing in cognitive sensor networks, and Wi-Fi and LTE networks.

The contributions of this article are fourfold:

- We formulate the problem of *adaptive sampling and low-cost monitoring*, focusing on event-sensitive data, and we design `e-Sampling` to address the problem.
- To make the best use of networks for diverse applications, we present a sensor embedded algorithm, which is adaptive to adjust the sampling rate autonomously and is computationally inexpensive, and it does not require neighbor's or sink's mediation.
- Based on the embedded algorithm, we present an event identification algorithm suitable for decentralized computing in energy-constrained networks. For this reason, we study SHM as a typical application of high-frequency events in the area of critical infrastructure protection and the complex FEM (Fire Event Monitoring).
- We demonstrate the effectiveness of `e-Sampling` in simulations with data traces collected by a real system on the Guangzhou New TV Tower (GNTVT) [Ni et al. 2009]. Also, we implement a prototype networked sensing system by using TinyOS [T2 on Imote2 2010] and Imote2 sensors [Crossbow Technology Inc. 2007], and deploy it on a specially designed physical structure to validate our algorithms. The results show that `e-Sampling` remarkably outperforms state-of-the-art schemes in terms of energy cost, system lifetime, and low-cost event monitoring.

This article is organized as follows. Section 2 presents motivation and related work. We offer problem formulation and models in Section 3. Section 4 briefly explains the design of `e-Sampling`. Section 5 presents the adaptive sampling algorithms. Section 6 provides a decentralized computing algorithm for event indication. Evaluation through both simulations and experiments is conducted in Section 7. Finally, Section 8 concludes this article.

2. MOTIVATION AND RELATED WORK

In this section, we first discuss the need of event-sensitive autonomous sampling adaptation from high-frequency event monitoring application perspectives. Then, we discuss the related work.

2.1. Networked High Data Rate Sensing Applications and Motivation

We describe some applications in more depth and the motivation of `e-Sampling`.

Acoustic-Based Monitoring. Acoustic observations enabled by dense sensor deployments can provide scientists with a deeper perception of wildlife interactions, building monitoring, and smart spaces [Luo et al. 2010; Mainwaring et al. 2002]. These applications collect raw acoustic waveforms sampled at more than 8kHz. Habitat monitoring is carried out on the Great Duck Island [Mainwaring et al. 2002]. Sensors are deployed in burrows of Storm Petrels (a sea bird) for monitoring purposes. During the daytime, the burrows are expected to be empty; a low sampling rate should be sufficient. However, if some events (i.e., unusual measurements) are recorded at some burrows, it would be desirable to collect samples from them more frequently (at a high rate) than other burrows and at any other time.

Environmental Monitoring. In flood monitoring, networks are deployed to monitor the water level in the river, lake, or the like [Kho et al. 2009]. The principle aim of such monitoring is to provide a warning to the user or the sink ahead of a flooding, such as 2 hours in advance, such that precautionary actions can be taken to alleviate risks to people and property. When a sensor node experiences that the water level is about to reach the warning level, it requires sampling more frequently (at a high rate). However, such warning level is a rare event. Monitoring the status of snow coverage also requires event-sensitive sampling. Snow is a mix of ice, water, and air, that can be measured at different frequencies in range {0.5, 100}kHz [Alippi et al. 2010]. As an "emergency alarming" application, fire event monitoring involves several sensing units. All of the units simultaneously acquire data and produce a huge amount of data. Handling all data at the sink induces transmission latency of alarming packets. Thus, sampling rate and interval should be adjusted according to the presence of the events (e.g., fire, air pollution, and snowfall).

Structural Health Monitoring (SHM). In SHM, detection of a physical event, such as damage/crack, is performed through the analysis of vibration signal characteristics [Liu et al. 2011; Li et al. 2010; Hackmann et al. 2014; Bhuiyan et al. 2014, 2015a]. SHM algorithms work on the measured data of multiple sensors generally via various matrix computations (e.g., eigen-decomposition, singular value decomposition [Nagayama et al. 2006]). The data from each sensor involved is no longer a single value but a sequence of data with length generally more than X0KB. Sensors capture at high rates (X00Hz to X000Hz), and acquire data continuously for a long period of time, even when there is no remarkable change in the frequency content of signal. Therefore, we argue that it is not always necessary to broadcast a huge amount of data over a network when there is no event. In such a case, we also argue that there is no need to further collect data at high rates.

Spectrum Sensing. A wireless cognitive radio network is allowed to reuse the frequency spectrum that is licensed to another system. Finding reuse opportunity needs dynamic sensing [Sobron et al. 2015]. In Wi-Fi and LTE coexistence, the fairness in the spectrum usage is low, where dynamic rate adaptation can be crucial [Yun and Qiu 2015].

Discussion. Any application of networks mentioned previously that requires collecting data at such high rates will clearly use a lot of energy, bandwidth, and other resources. Energy-limited deployments should sample at high rates only occasionally—with duty cycling, say, in the presence of events of interest. This makes a reduction on network resource wasting.

2.2. Related Work

Adaptive sampling has been a resource management issue in networks [Hao et al. 2015; Aderohunmu et al. 2015; Alippi et al. 2010; Wang et al. 2010b; Burer and Andlee 2007]. When monitoring an event using a network, a high accuracy should always be guaranteed. This can easily be achieved by using naive solutions of sampling at a higher rate than necessary for the representation of the signal bandwidth. Several network applications motivate our work, each of which collects data at a high rate. Small-form-factor, low-power wireless sensors are convenient to deploy, but mote-like sensor devices, such as the relatively advanced TelosB and MICA2, cannot continuously transmit at the rates these applications require.

Optimal sampling in networks focuses on how to assign the sampling rate under given bandwidth constraints [Shu et al. 2008] or under conditions of minimum energy usage. The closely related work, called FloodNet [Kho et al. 2009], provides the application-layer design of networks, where the sampling rate is adjusted according to the estimation error or regression accuracy of the physical phenomenon measured [Kho et al. 2009]. It maintains an acceptable signal reconstruction at the sink to detect an event. The data collection at a fixed period of time such as an hour and adjustment of sampling rates based on the analysis of a large amount of data, consumes a significant amount of energy. Such an adjustment may not be applied to many dynamic or high-frequency events (e.g., an event of fire, damage/crack, or an earthquake, a physical activity); a sensor cannot adjust its rates as such an event occurs for a short period of time (e.g., 1 second, 5 seconds, or 5 minutes). However, it has serious limitations on the quality of signal frequency, which may vary largely from time to time and also change suddenly with dynamic events [Masoum et al. 2012].

Sensor placement and selection of sensor nodes as spatial sampling problems is proposed in Wang et al. [2010b], which is an asynchronous sampling strategy based on randomly assigning sampling shifts. In another study [Burer and Andlee 2007], an optimization problem corresponding to maximum entropy sampling is formulated, which proposes a solution to the problem, using factor masks. The utilization of application-specific filtering for data reduction before transmission is effective for high-rate applications [Greenstein et al. 2006]. However, the filter behavior is hard to predict, and applications perform poorly if filtering is too aggressive or poorly calibrated. Backcasting is a prominent method that operates by activating only a small subset of nodes that communicate their information to the sink [Willet et al. 2004]. This provides an initial estimate of the sensed environment, and guides the allocation of additional network resources. The sink then selectively activates additional sensor nodes in order to achieve a target error level based on approximating signal activity. However, the nodes are allocated at a fixed bandwidth and sampling rate, where they should keep long-range communication links with the sink. Waiting for the allocation from the sink from time to time, and/or periodically, is expensive.

Compressive Sensing (CS) techniques have been used to allow the reduction of the sampling rate for specific signals (e.g., images) [Hao et al. 2015]. Existing work in CS for networked sensing uses fixed sample rates, which may make a network unable to capture significant dynamic changes in certain application environments (including damage, crack, fire, physical activity) unless the rate is sufficiently high. This degrades the sensing quality. A recent scheme to pursue high sensing quality at low sample rate used an adaptive CS-based sample scheduling mechanism (ACS) for networks [Xu et al. 2015]. ACS estimates the minimum required sample rate subject to given sensing quality on a per-sampling-window basis and accordingly adjusts sensors' sample rates. Estimating sensing quality based signal acquisition and the rate adaptation may not be useful dynamic rate adaptation. However, it shows high computational cost.

A Quality-aware Adaptive Sampling ("QAS" for short) algorithm is suggested in terms of energy consumption and data quality [Masoum et al. 2012]. In the algorithm, when a node experiences stability in its environmental condition, it reduces its sampling frequency. By doing so, the number of data transmissions between the nodes and the Cluster Heads (CHs) are assumed to be reduced. Both the nodes and the CHs employ the same prediction model, which describes the given environmental attributes. Utilizing QAS, every node transfers data to the CHs only when an environmental condition is not stable and thereby its data prediction is not accurate enough. In fact, the sampling rate adaptation requires a lot of data transmission in each cluster and puts a burden on a CH. Moreover, it is also difficult to provide a specific sampling rate to a region of interest, where an event occurs. In the evaluation, we found that, once the sampling rate becomes high, in the absence of an event, achieving the lowest sampling rate is not possible in practice in these schemes.

Applying the preceding methods, if one wishes to analyze event information at the sink, this mechanism is not suitable for some real-time applications, especially for high-rate and emergency alarming applications (e.g., the event of a fire or damage, intrusion detection, road patrolling, etc.), where the alarm is stringently required to be announced with very low latency. The drawbacks of most of the existing sampling methods are that the sampling method is not fully autonomous and adaptive, and the real-time requirement is not taken into account. Enabling nodes to rely on the neighbors, the CH, or the sink to assign or estimate their rates requires a lot of packet transmission separately and the adaptation is not real-time, thus inducing network latency and energy cost in each round of monitoring.

A noteworthy work related to `e-Sampling` depends on a graph-based frequency analysis for event detection [Shuman et al. 2013; Ciancio et al. 2006]. The graph Fourier transform and related Fourier analysis are carried out. The idea of low to high frequency of events may hold up in this graph paradigm, where events can be monitored in a sensor network, which can be represented as a graph. Applying such a concept in various applications may provide high performance in event detection. However, it may need an important portion of resource consumption and detection delay due to sampling rate adaptation and correlation analysis in the graph. One reason is that, in the graph-based analysis, sampling algorithms are employed to detect events at a high, low, or fixed frequency sampling, which should be adapted by a central unit. In `e-Sampling`, the spectral analysis and sampling rate are in the classical sense, not the graph-based [Shuman et al. 2013].

The key aspect that differentiates `e-Sampling` from the prior efforts lies in both data acquisition and decision making. `e-Sampling` allows an ongoing estimate of frequency content. Sensors adjust sampling rates independently, and do not wait for the sink or neighbors' interruption and make a decision in the absence/presence of an event.

3. PROBLEM FORMULATION AND MODELS

Consider a set S of N sensor nodes given to monitor events in a data-intensive network application. Assume that they are deployed by some generic deployment strategy (such as uniform, random, or deterministic [Li et al. 2010]) at feasible locations $L = \{l_1, l_2, \ldots, l_N\}$, where sensor i is placed at location l_i, and l_0 is a suitable location of the sink. Let X be the communication range, where the maximum and minimum communication ranges of a sensor are X_{max} and X_{min}, respectively. X_{min} is used to maintain local topology, where every pair of sensors within X_{min} is allowed to share and compare its decision with its neighbors. Sensor i corresponds to a node, and any two nodes are connected if their corresponding nodes in X_{min} can communicate directly. The intention of adopting adjustable X is to reduce energy costs for frequent long distance transmission. Advanced sensor platforms, such as Imote2, support discrete power levels [Crossbow Technology Inc. 2007].

Sensors are allowed to collect data, process locally, make a decision on physical event indication, and transmit the decision directly to the sink. Particularly, given proposed algorithms, each sensor should calculate their sampling rate based on its own collected information. That is, a sensor's sampling rate computation does not rely on its neighboring nodes or network graph. The sampling rate computation is fully autonomous. Then, the event detection based on a high or low rate sampling is also fully autonomous. A sensor can adapt its sampling rate based on its need at its location/environment. It does not need a neighboring sensor's sampling rate adaptation or situation. No correlation is assumed between sensors. Once a sensor makes a decision and finds that its information is relevant to the presence of event information, it then shares the decision with neighbors to ensure whether or not there is an event.

3.1. Energy Cost Model (E_i)

One objective is to minimize network energy cost, hence to maximize the network lifetime. We achieve it by reducing the total energy cost, denoted by E_i, on a sensor i in different aspects, including sampling, Analog to Digital Conversion (ADC), computation, and communication. First, we briefly describe how energy is consumed by a sensor communication component in packet transmission/reception. The maximum energy cost of a sensor depends on a routing protocol used by the data collection application. This falls into the domain of power-aware routing [Olariu and Stojmenovic 2006; Singh et al. 1998]. Consider a routing algorithm [Singh et al. 1998]: let $q = z_0 z_1 \ldots z_k$ be the path from a sensor to a neighbor in X_{min} or to the sink; we define $q[i]$ as the ith hop sensor on the path q, and γ_q as the amount of traffic flowing along path q within each round of monitoring data collection. Then, $\overline{q[i]q[i+1]}$ is the distance between any two sensors $q[i]$ and $q[i+1]$. $\overline{q[i]q[i+1]} \leq X_{min}$ is used for data delivery to the neighbors and $\overline{q[i]q[i+1]} \leq X_{max}$ is used for data delivery to the sink. Thus, E_i is decomposed into the following parts:

$$E_i = e_t + e_{comp} + e_{ADC}. \tag{1}$$

(i) e_t is the total energy cost per bit for transmission over a link between a transmitter and a receiver, that is, e_s and e_r are the energy cost for receiving and transmitting data, respectively. Hence,

$$e_t = \sum_{\forall q, \exists i, q[i]=l_i} \gamma_q \cdot e_s(\overline{q[i]q[i+1]}) + \sum_{\forall q, \exists i, q[i]=l_i} e_r(q[i]). \tag{2}$$

We do not consider the distance between two nodes when calculating energy cost for receiving data. The energy cost of a node for reception is distance-independent.

(ii) e_{comp} is the energy consumed by the computation that is mainly due to the onboard processor, such as a microcontroller, digital signal processing (DSP) chip, or Field-Prgrammable Gate Array (FPGA) [Gutnik and Chandrakasan 1997]. These devices consume energy proportional to the number of processing cycles, as well as the maximum processor frequency f, switching capacitance μ, and hardware-specific constants k and β, respectively [Gutnik and Chandrakasan 1997]. The number of cycles required to perform a task on the amount of samples (m) are estimated according to the computational complexity $O(m)$, which describes how many basic operations (i.e., averages, additions, multiplications, etc.) must be performed in executing the task. The computational energy to complete a task can be calculated according to

$$e_{comp} = O(m) \cdot \mu \left(\frac{f}{k} + \beta \right). \tag{3}$$

(iii) e_{ADC} is the energy consumed by the ADC. In the sampling, two of the modules are the most important, namely, the ADC and the sensor itself, when they need energy. As in most cases, if the samples come at fixed time intervals, the average energy can be related to the energy per sample and the number of samples acquired. However, in the case of event-sensitive adaptive sampling in `e-Sampling`, the energy cost can vary due to m and R_c (for symbol description, refer to Table I). R_c can be either R_h or R_l, as it is set and can be stable for an estimated time after selection of a sampling rate. Thus, e_{ADC} of the ADC is typically proportional to m and the sampling rate used [25 html; Gutnik and Chandrakasan 1997].

$$e_{ADC} = \frac{Q_1 \cdot R_c + Q_2}{R_c} \cdot m, \tag{4}$$

where Q_1 and Q_2 are the gate capacitor constants of the ADC [Texas Instruments 2013].

Table I. Symbol/Variable Description

Symbol	Description
F_h	high-frequency content of input signals
R_h	sampling at a high rate
R_l	sampling at a low or adjusted rate
R_m	minimum required sampling rate
R_c	current sampling rate (i.e., continue activities at this rate)
d	time interval index, $d = 1, 2, \ldots$
D_h	the duration of each burst of high-rate sampling
D_l	the duration (short interval) of low-rate or adjusted sampling
I	given a whole monitoring round or time interval

We calculate the remaining energy (E_i^{rem}) reserved for sampling on sensor i at the beginning of a given monitoring round I. Let $E_i^{req} = \max_{i=1,2,\ldots,N} E_i$, that is, E_i^{req} be the maximum energy required on i (which is equivalent to E_i) for a set of a actions in I. We define the system lifetime T to be the total number of rounds of monitoring data collection before any battery runs out of energy:

$$T = E_i^{rem} / E_i^{req}. \tag{5}$$

3.2. Problem in `e-Sampling`

Given a set S of N sensors for monitoring events, $S = \{s_1, s_2 \ldots s_N\}$. Each sensor $s_i \in S$ has a sampling actions it can perform, and these actions are denoted as $A_i = \{R_i^1, R_i^2, \ldots, R_i^a\}$. In the adaptive sampling context, these actions represent sampling rates that i opts to perform at any particular point of time within T, and adjusts its D_l and D_h. Therefore, a sensor can only select one action at any particular point of time and continues for the duration of D_l or D_h; however, it can then choose a different R_i^x at subsequent selection points. This means that s_i is allowed to adjust its actions by analyzing its recent samples of F_h and the samples in F_h and the set of signals that it believes it will observe.

The following are the key constraints:

—*Action selection constraint*—a sensor i can only select one sampling action at any particular point of time.

—*Energy constraint*—a sensor i requires a certain amount of energy E_i^{req} to take a sampling action in I that must not exceed E_i^{rem}, that is, $\sum_{o=1}^{a} R_i^o E_i^{req} \leq E_i^{rem}$.

—*Application-specific constraint on network data-collecting ability*—This represents the total information collected by the entire network. *e-Sampling* does not need to know about the event type in a specific application during the sampling rate adaptation but during data processing for monitoring the events. In different applications, different types of events have different signal intensities. There are also some events that are composite, such as, snow, fire, or air pollution. Thus, the data-collecting ability may vary from application to application.

—*Computational complexity*, $O(m)$.

Key objectives are to minimize $\sum_{i=1}^{N} E_i$ and to maximize T.

4. THE DESIGN OF e-SAMPLING

We focus on event-sensitive adaptive sampling or more general techniques focusing on optimizing data acquisition. Adjusting (taking actions on) sampling rates or putting desired SR into action according to the frequency content of the input signals is an intuitive approach. See Figure 1 for frequency content (F_h). However, it is challenging to achieve in practice, which is due to the fact that the presence of a physical event

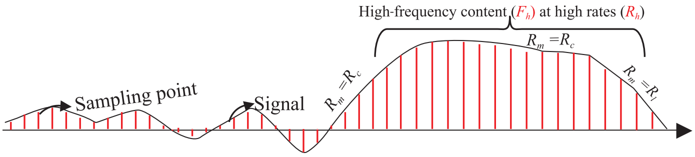


Fig. 1. The concept of high-frequency content F_h of `e-Sampling` collected at high rate.

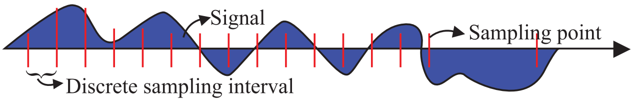


Fig. 2. Illustration of one-dimensional signal indicating, sampling interval, sampling points, and how to sample it.

ALGORITHM 1: e-Sampling in Each Network Node

```
DecentralizedControl{                    //1st stage data reduction
  While (True) {
    S.RateComp in $D_h$ = True{
      // start Sampling Rate Computation at the beginning of
         the system or at a certain interval $I$
    Run Algorithm 1
       // Sampling rate and interval adaptation
    Compute new $R_c$ // set a new sampling rate
    Compute $D_l$}}} // set the duration for the new rate
    ComputeEventIndication{                  // 2nd stage data reduction
    Run Algorithm 3 // the Event Indication Algorithm
      If indication.Strength $\geq$ 40%
         transmit the indication
      else transmit an acknowledgment}
```

is dynamic or unknown until after sampling. During a monitoring, a possible event contains a set of high-frequency content that is the actual data acquired from the sensors.

In the beginning of an interval, sensors start short and recurrent bursts of sampling at a high rate (R_h), and examine these samples to analyze F_h. The sensor probes the entire bandwidth, which is the opposite of checking only the bandwidth visible at some sampling point (see Figure 2) when sampling at another time at a low/adjusted rate (R_l). D_h is the duration of each burst of sampling at R_h that is followed by D_l, where the sampling rate used here is calculated based on findings in F_h. The time between two neighborhood sampling points is called a “discrete sampling interval.”

Whenever an event occurs, F_h becomes important (i.e., the changes in this F_h are large, which implies that there may be an event) and D_h is long enough, as shown in Figure 1. Thus, the sampling rate is kept at R_h until F_h is unimportant. Upon analysis, if an event is detected at a sensor node in this interval, this event is called *high-frequency event*. Once F_h becomes unimportant, D_l shortens in this interval. Before F_h is known, D_h is followed by a relatively long D_l. Upon analysis, if an event is detected at a sensor node in this interval, this event is called *low-frequency event*.

Once F_h becomes unimportant, the sampling condition is again relaxed: D_h shortens and D_l lengthens. Thus, D_h and D_l, and R_h and R_l are automatically switched, depending on F_h. Using this technique in event detection, an analysis of F_h is preserved in each discrete interval and, subsequently, a better sampling rate is selected, while reducing false negative detections. This reduces the energy cost of sensors in almost all aspects (e.g., sampling, ADC, computation, and transmission), and therefore longer lifetime. To the best of our knowledge, this is the first decentralized computing or control approach that attempts to adjust the optimal sampling rate by capturing the highest frequency content and detects the possible situation based on the frequency content. This means that in this system, each sensor adjusts its sampling rate to an optimal rate and provides a damage indication locally without any or neighbor central intervention.

The procedures of `e-Sampling` are simply shown in Algorithm 1, and are executed by each sensor node individually. `e-Sampling` reduces the amount of data in two stages: during the period of sampling (lines 1–9), and during the period of decision-making on an event (lines 10–14). For event monitoring, a decentralized computing algorithm is used to make decisions on the absence or presence of events of interest based on F_h. Both stages are performed at individual sensors in a decentralized manner. The ADC task is performed within an interrupt routine, and the main program performs the sampling rate selection computation. At the system initialization, the sampling is carried out at R_h. When enough samples are acquired, “DecentralizedControl” is executed to select an appropriate sampling rate. As the computation of the rate selection is over, duration D_l starts, after which the new adjusted sampling rate is used in D_l. Data is stored after each sample is taken in both periods. All of the sets of F_h are stored in the sensor local memory (or flash memory). A sensor keeps them within the memory until it receives a confirmation message from the sink, or until the memory is full. After each complete sampling period I, a sensor calculates an event indication.

5. ADAPTIVE SAMPLING: FIRST STAGE DATA REDUCTION

In this section, we discuss the adaptive sampling technique.

We propose the autonomous adaptive sampling algorithm (Algorithm 1). In the algorithm, in line 1, a sensor first starts acquiring samples at a high rate, and stores them into the buffer. We need to discuss the sampling interval: how to adjust the interval for a duration in which a sensor takes samples at a high or low appropriate sampling rate. For both sampling rate and interval adaptation, each sensor acquires data in d-th given time interval. Each of such intervals consists of two subintervals:

(1) D_h starts with a short investigative subinterval in which an R_h is adopted.
(2) D_l (the remainder of the interval in each time interval) starts when the sampling rate is adjusted to a lower rate, based on the frequency content, which is influenced by the presence of an event.

Thus, the adjusted R_l for D_l is adopted based on the required rate, R_m. D_l (the remainder of the interval in each time interval) starts when the sampling rate is adjusted to a lower rate, based on the frequency content, which is influenced by the presence of an event. For example, see Figure 3 for how frequency content is influenced by a particular event. We analyze samples from our experimental scenario in this case. We have a test structure with a set of sensors deployed to observe SHM performance. When we perform hammer strikes at some points of time on the structure, the frequency content F_h is changed at the points of time. Figure 3 analyzes the influence by two-time hammer strikes. These strike events lead to a maximal sampling rate from 1100 to 1400Hz (more details are given in the evaluation section). Storing the information about the rates R_m, D_h, and D_l temporarily along with acquired signals are illustrated in Figure 4. This result corresponds to our main concept in Figure 1.

An analysis is performed on the data points acquired during D_h to estimate the highest frequency content F_h. After analyzing, the sensor executes line 6. If the sensor discovers that F_h is important, then the minimum required rate is the high rate, that

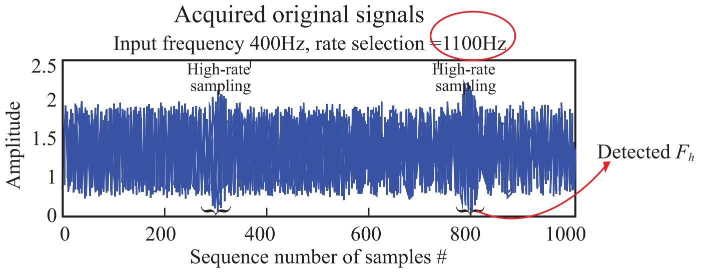


Fig. 3. Representative example: how frequency content is influenced by a hammer strike in the case of SHM.

ALGORITHM 2: Autonomous Adaptive Sampling (Rate and Interval Adaptation)

```
1:  for sensor $i$ in $S$
2:    At some $d$-th interval
3:      Sensor $i$ starts data acquisition
4:        Get samples at $R_h$ for $D_h$
5:          Store into buffer
6:          Compute rate, $R_m$
7:        Compute $R_c$ via Algorithm 3
8:          Store into buffer // see Figure 5
9:        Read buffer for $R_l$
10:         Adjust $R_l$ based on $R_c$ if needed
11:       Get samples at $R_l$ for $D_l$
12:       Store the samples into buffer
13:   $d = d + 1$
```

is, $R_m = R_h$. It is determined as

$$R_m = c \cdot F_h, \tag{6}$$

where $c \geq 2$ is a confidence factor chosen to satisfy Shannon's sampling theorem [Alippi et al. 2010]. By Shannon's theorem, the sampling rate should be at least twice that. However, F_h is generally unknown before sampling, due to signal activity or the absence/presence of an event. To know this, lines 7–10 are executed of that of an embedded algorithm (Algorithm 3), which helps to get the current rate, R_c, and to make a decision whether or not the current rate should be continued or adjusted. If F_h is still important (there is possibly an event), the sensor continues sampling at the current rate, $R_m = R_c$; otherwise, it adjusts the rate to a lower rate, R_l. After getting the lower rate, the sensor continues sampling at this rate (lines 11 and 12), until the next d-th interval.

For increased robustness, R_h or R_l is estimated by taking into account the sampling rate in the k previous intervals:

$$R_c = \max(R_m(d - j)), \quad j = 0, \ldots, k - 1, d \in I. \tag{7}$$

For simplicity, the sequence of operations in adaptive sampling, intervals, subintervals, and resulting dynamic sampling rate are presented in Algorithm 2. In the d-th interval, either D_h or D_l can be zero (see right plot of Figure 4). According to this discussion, we need to select an appropriate sampling rate. Thus, D_l must also be selected, which governs how often the sampling rate is updated.

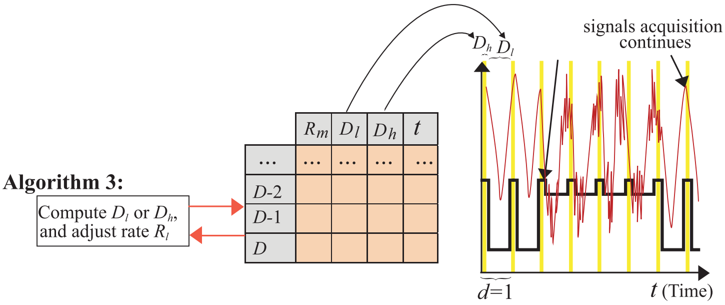

Algorithm 3:
Compute D_l or D_h, and adjust rate R_l
R_m
D_l
D_h
t
...
D-2
D-1
D
D_h D_l
signals acquisition continues
d=1
t (Time)

Fig. 4. Example of temporarily storing R_m, D_h, or D_l, and the time each subinterval occupies during Algorithm 3 execution.

5.1. Sampling Rate Selection

In Algorithm 2, the analysis on the samples of F_h helps compute a choice of sampling rate in a fast and efficient manner. Here, a key technical challenge is to estimate the proper sampling rate based in the pretense of an event. This is challenging because when there are changes in the environments due to the physical event, the sensor acquired signals collected at both high rate and low rate reflected by changes are mixed. Our key observation is that these signals are linearly combined so that their frequency content samples are preserved when they are mixed together. Therefore, we need to split the samples of high rate from the samples of low rate. To achieve this, various signal processing tools can be used, including Wavelet Packet Decomposition (WPD), Short-Time Fourier Transform (STFT), Principal Component Analysis (PCA), and so on.

The most relevant signal processing tool that can enable us to extract high-frequency content samples at multiple resolutions is WPD. The advantage of Discrete Wavelet Transform (DWT) is that it provides a proper trade-off between time and frequency resolution and enables the measurement of both fast and slow, or high or low signal activities. Another advantage is that WPD is desired to be power complementary, meaning that the summation of their responses is equal to one over the frequency domain. Wavelets has as few vectors as possible. DWT calculates the energies in different levels at any given time, where each level corresponds to a frequency range.

5.1.1. Wavelet Packet Decomposition for Signal Splitting. We split the acquisition of samples into frequency bands at a sensor's current sampling rate to estimate F_h. WPD technique splits the samples and separates the high-frequency content samples that may represent event information. WPD is leveraged to break the signal into frequency bands, as shown in Figure 5.

In the WPD, during the decomposition procedure, the initial step splits the original signal into two parts, that is, it decomposes the set of samples into separate frequency bands by recursively applying high-pass ($H(z)$) and low-pass filters ($L(z)$), where $H(z)$ and $L(z)$ are the z transforms of Finite Impulse Response (FIR) filters $h[n]$ and $l[n]$ [Mitra 2005]. A complete subband decomposition can be viewed as a decomposition of the acquired signal, using an analysis tree of depth, $\log r$. We illustrate a tree-based WPD on the acquired signals in Figure 5. Figure 5(a) shows two-level packet decomposition, while Figure 5(b) shows multilevel packet decomposition based on the filtering. The filters are desired to be power complementary, meaning that the summation of their responses is equal to one over the frequency domain. This is why we choose wavelets that have as few vectors as possible [Alippi et al. 2010; Khan et al. 2013; Keinert 2004]. The effect of scaling functions in wavelets is a kind of filtering

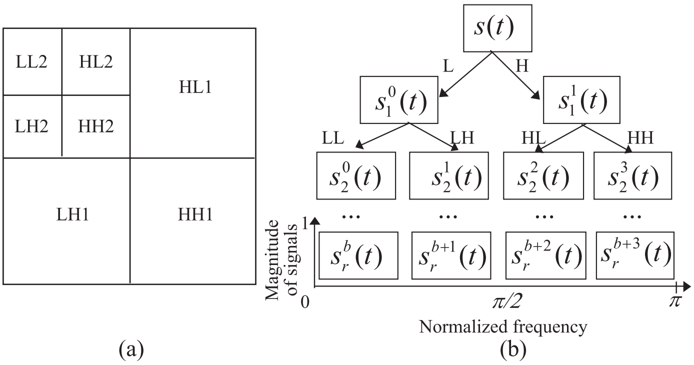


Fig. 5. The two-dimensional wavelet packet decomposition: (a) two-level decomposition; (b) multilevel decomposition.

operation on signals, which acts as a low-pass filter, screening out F_h. This ensures that all signal information is being examined equally.

In Figure 5, the decomposition can continue downward multiple levels r, generating $b = 2^r$ frequency subbands. These indexes r and b are exploited to distinguish each subband in the wavelet packet tree, $s_r^b(t)$. The low-pass and high-pass filtering operations, which generate the subbands, are explained mathematically as discrete convolutions:

$$\begin{aligned} s_{r+1}^{2b}[n] &= \sum_p h[2n-p]\; s_r^b[p], \\ s_{r+1}^{2b+1}[n] &= \sum_p g[2n-p]\; s_r^b[p]. \end{aligned} \tag{8}$$

Note that the signal is decimated by a factor of 2 after each filtering operation, indicated by $2n$ in each equation. This means that half of the coefficients are discarded after filtering, and the total number of coefficients in all bands at each level r is about the same for each level. This decimated wavelet packet formulation enables greater computational efficiency by keeping the number of coefficients about the same for each level of decomposition, although the number of bands increases by a factor of 2 [Saito and Coifman 1995]. This decimation is made possible because each filtering operation essentially halves the bandwidth of the previous band, allowing half the samples to be removed without loss of information.

However, the filters are never perfect half-band filters. Therefore, proper construction of the filters is critical to truly represent the signal in these bands, due to the aliasing effect induced by decimation, and the slightly overlapping frequency responses as illustrated in Figure 5(b). Specifically, filters should be selected as the decomposition filters of a Quadrature Mirror Filter (QMF) bank [Jairaj and Subbaraman 2010]. These are two out of a set of four filters: two decomposition and two reconstruction filters, which meet the criterion of being power complementary, as well as canceling out any aliasing due to decimation [Mitra 2005]. For reconstruction , at each level the series $s[n]$ is transformed into another of the same length, that is,

$$\begin{aligned} \{s[n]\} &\rightarrow y[n] \\ &= \left\{ \sum_p s[p]h[n-p]; \sum_p s[p]l[n-p]; \right\}. \end{aligned} \tag{9}$$

5.2. Threshold and Estimation of Sampling Rate

The filters break the signals into 2^r frequency bands, as previously discussed using the WPD, as shown in Figure 5. Once the bands are generated for some r level decomposition, where there will be 2^r bands, coefficients in all bands are subjected to a threshold ω, which is a function of the data size m, noise variance σ, and error constant τ, taken from the first high-pass band as

$$\begin{aligned} \omega &= \sigma\sqrt{2\log m}, \\ \sigma &= \frac{median(\{s_1^1(W)| : \forall W \in s_1^1\}}{\tau}. \end{aligned} \tag{10}$$

Any coefficient not above ω is set to zero and $\tau \in [0.5, 1]$ [Donoho and Johnstone 1994]. Now, the highest band index b with any nonzero coefficient, denoted by b_h, is identified, indicating an estimate of F_h of signals. We utilize this index and compute an appropriate sampling rate, called a control rate, for the signals as follows:

$$R_m = c \cdot \frac{R_h}{2^r} \cdot (b_h + 1). \tag{11}$$

Here, $c \geq 2$ [Alippi et al. 2010]. We only need to identify one band, that is, the highest band with any content in it, b_h, as the computational load induced by determining an appropriate sampling rate. Algorithm 2 completes this task by selectively computing bands; it does this by computing only two bands at each r instead of 2^r bands at each r. If there is F_h in the high-pass band, the high-pass band is passed on and split again. If there is no such F_h, the low-pass band is further split. As a result, the WPD tree is allowed to be traversed; only computing two bands at r, and completely avoiding computing unnecessary branches, the computation time is $O(m)$.

By using Algorithm 3, an appropriate sampling is identified, that is, R_c is either R_h or R_l; the sampling rate is adjusted, or lowered, to this rate in each discrete interval. By implementing this technique, computational complexity is now linearly related to the number of samples needed to be processed, which has much lower computational complexity than the Fast Fourier Transform (FFT) algorithm. The computational

ALGORITHM 2: (Embedded): Sampling Rate Selection

```
if there an update of sampling rate then
  $U = 0$
  $s_0^0(t) = s(t)$
  While $r < R \&\& U = 0$ do:
     {
      Compute Equation (7)
      Compute $R_c$ via Algorithm 3
      if  (there is $F_h$ in $s_{r+1}^{2b+1}(t) > \omega$ for any $t$
        $s_r^b(t) = s_{r+1}^{2b+1}(t)$
       there is possibly a situation of the presence of a physical event)
       elseif $s_{r+1}^{2b+1}(t) > \omega$ for any $t$
         $s_r^b(t) = s_{r+1}^{2b+1}(t)$
       //there is possibly a situation of the presence of a physical event
       else $U = 1$
     }
     $b = b_k$
     Compute $R_m$ //using Equation (10);
     Set $R_c = R_m$
     end while
```

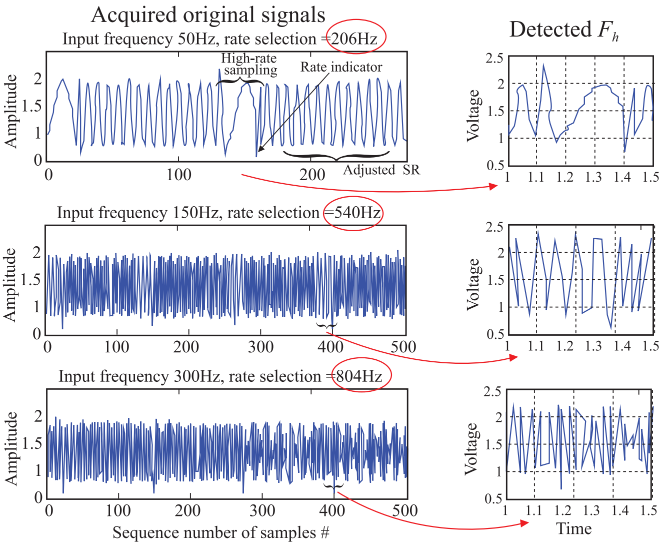


Fig. 6. Autonomous sampling rate adaptation of the fourth sensor node.

complexity in actual filtering becomes only that of the discrete wavelet transform, as approximated by

$$O(m) \approx \sum_{r=0}^{R} 2\left[\frac{m}{2^{r-1}}Q + Q(Q-1)\right]. \tag{12}$$

Here, m is the size of the dataset, and Q is the length of the QMF filter chosen. This result is obtained by counting the basic operations needed to perform the filtering, such as addition and multiplication. The benefit of selectively computing wavelet coefficients as opposed to computing the full WPD when calculating a rate is analyzed in Figure 5. This compares favorably with the FFT algorithm, which requires $O(m \log m)$.

Figure 6 analyzes the results on the acquired vibration signals by a sensor in our experiment, each of which is a sine wave in the range of F_h (left-hand plots). It is seen that for each signal, the correct F_h and sampling rate is selected by the algorithm. After analyzing the set of F_h at different selected rates (circle marked) from the acquired samples, important F_h in the periodic stretched out portions indicates D_h (right-hand plot). Each D_h portion is followed by 0 and 255 values, which look like a spike, followed by the sampling rate adaptation. It is seen that for each signal, the correct F_h and sampling rate is selected by the algorithm.

6. DECENTRALIZED COMPUTING EVENT INDICATION: SECOND STAGE DATA REDUCTION

This section describes sensor decentralized computing in order to provide an indication of the absence/presence of events. In e-Sampling, sensors generally do not need to know the types of events during the adaptation of sampling rates and intervals. We expect that e-Sampling can be applied in a wide range of high-data-rate applications, as described in Section 2.1 However, in different applications, they need to know application-specific adaptation, since detection of different types of events requires different sampling rates and intervals.

In the case of flood monitoring, the detection of a flood warning level is a rare event. In snow monitoring, status of snow coverage also requires event-sensitive sampling. Snow is a kind of composite event (i.e., ice, water, and air) measured at different frequencies in {0.5, 100}kHz. Fire is also a composite event that requires very high-frequency sampling. Physical activity monitoring is another high-rate application that may require switching components into sleep mode between samples, or adjusting the sampling rate to a lower rate during nonphysical activity. Any network applications, like previously, that acquire data at high rates, will clearly use a lot of energy, bandwidth, and other resources. As representative applications from the area of critical infrastructure protection and environmental monitoring, we consider monitoring events in two data-intensive schemes: SHM application and Fire Event Monitoring (FEM). Here, we are going to apply our sampling algorithms for SHM applications. The FEM application is described in the Appendix.[1]

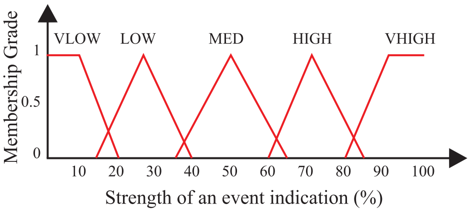
6.1. SHM

Despite being an effective scheme to assess with precision the monitoring of physical environment such as structural environments, transmitting all of the data collected by the sensor nodes to the sink node for ofline structural mode drains a consistent amount of energy and requires a long time, particularly in the case of a tree-like multihop network [Bhuiyan et al. 2014, 2015c]. WSNs have long been used by civil or structural engineering domains for SHM. In a typical SHM system, the objective is to monitor structures (e.g., buildings, bridges, aircrafts, nuclear plants, etc.) and to detect possible events (e.g., damage, crack) of them at an early stage, which prevails throughout the engineering domains. SHM algorithms work on data acquired at a high fixed-rate and let sensors work for a fixed period of time, say from 10 minutes to hours or days [Hackmann et al. 2014]. The algorithms are carried out globally by the sink to identify events. This global monitoring, with handling a large amount of data transmission, brings difficulties to networks.

Based on our experience obtained from collaborations with civil researchers, the general concern is whether the network-based SHM system can replicate the data delivery functionality of the original wire-based counterpart, which may have less interest in addressing the constraints of the networks, and embed in-network processing algorithms. Therefore, the method of data acquisition in traditional SHM [Li et al. 2010; Kim et al. 2007] can be improved by our sampling algorithms. `e-Sampling` presents a network in which the nodes are equipped with some sensing units for SHM. We consider a three-axis accelerometer that acquires vibration data, caused by ambient/forced excitation. After data reduction in the first stage, it is still infeasible for a resource-constrained sensor to transmit a set of F_h. We use the sensor decentralized computing to provide an indication shortly based on F_h, that is, closely analyzing those ranges of samples to find significant changes.

All of the signals collected under the adaptive sampling rate are marked by the D_l, D_h, and time stamp. A sensor has all of the acquired samples sequenced in the database. Each sensor is designed to maintain a local database. We utilize a query-interface-based database for wireless sensor [Tsiftes and Dunkels 2011], by which all of the samples are aligned. A sensor first distinguishes the samples in F_h and then reads them from the database. Thus, only the samples of F_h are processed to extract an indication of the absence/presence of an event. The indication decision is normalized and graded, as shown in Figure 7, as the percentage of the strength of the event detection, as follows: $0\% \leq \text{VLOW} \leq 20\%$, $21\% \leq \text{LOW} \leq 40\%$, $41\% \leq \text{MED} \leq 60\%$, $61\% \leq \text{HIGH} \leq 80\%$, and $81\% \leq \text{VHIGH}$. If the strength is more than expected (e.g., "MED" (medium)), a sensor makes a pairwise comparison by sharing it with the neighbors (as shown in Figure 9) in X_{min}. If the strength is less than that, a sensor does not transmit the indication to the sink (refer to dataflow in Figure 9). In this way, if there is no damage indication, the

[1] The Appendix can be found in a supplemental file and also in an online link.



(c)

Fig. 7. Membership function: strength of an event indication.

amount of data is reduced before transmission. We utilize such a query-interface-based system suggested in Tsiftes and Dunkels [2011] considering communication constraint. The FTSHM system is based on the indication, because we think that damage is a kind of event that rarely occurs in a structure.

Let q^{Cur} be the current condition of monitoring the environment (e.g., physical structure). The following three equations are iteratively executed by each sensor:

$$\begin{cases} q^{Cur} = \eta \cdot q_1 - q_2 + \sum_{h=h_1}^{h_n} F(h), \\ q_2 = q_1, \\ q_1 = q_0, \end{cases} \tag{13}$$

where $F(h)$ is the recent set of samples of $F_h, h = 1, 2, \ldots n$; thus, the range of frequencies acquired at R_h is $[h_1, h_n]$. q_1 and q_2 store the results of the two previous iterations, which have been acquired at either R_h or R_l. η is used in the iterations as the application-specific data collection coefficient. We define the embedded indication function denoted as $\Delta s_u(e)$ to ensure that there is an absence/presence of an event in the vicinity of a sensor u. Let Ref be the reference sample set of F_h that has been acquired when there is an absence of an event in the application. $\Delta s_u(e)$ is computed by

$$\Delta s_u(e) = \frac{|q^{Cur} - q^{Ref}|}{q^{Ref}}. \tag{14}$$

Assuming that a sensor u has $n \in N$ neighbors in X_{min}, we normalize the output of $\Delta s_u(e)$ and put the output into the grade. It should be according to its membership grade shown in Figure 7. In this work, a sensor u shares its $\Delta s_u(e)$ with the n sensors if the strength is MED. It can be set as needed by the user. By comparing the strength among all pairs $(u \leftrightarrow v)$ of sensors in X_{min}, it is possible to correctly locate the event.

To allow a sensor to make a decision whether to transmit the indication to the sink or not, we define a transmissibility function given in Equation (14). The function is given by

$$F_{s_u}^{s_v} = \frac{\sum_{v=1}^{n} (\Delta s_v(e))^2}{\sum_{u=1}^{n} (\Delta s_u(e))^2}, \quad n \in N. \tag{15}$$

We also normalize the output of $F_{s_u}^{s_v}$ to 100% as the strength of the event indication, which is a confirming decision. If the strength is MED, a node transmits the indication to the sink. This also can be set by the user. The use of these decentralized computations prevents the nodes from transmitting a long sequence of acceleration signals (usually up to 60KB of data for each sensor node) to the sink node for ofline analysis.

6.2. Dataflow in the Network for Event Indication

In this subsection, we discuss the application dataflow, which is in two cases: (i) `e-Sampling-C`: centralized computing/monitoring including adaptive sampling

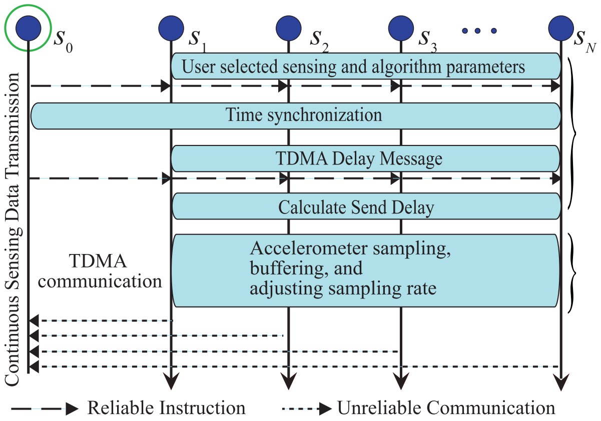


Fig. 8. Application flow in the case of centralized computing.

algorithms—where all of the raw data needs to be transmitted to the sink; (ii) `e-Sampling`: decentralized computing/monitoring as described earlier.

6.2.1. Centralized Computing/Monitoring. This includes adaptive sampling algorithms and excludes computation at the node level, that is, all of the raw data needs to be transmitted to the sink. Figure 8 depicts the overall application dataflow in the centralized computing case. In the beginning, the system user inputs the various sensing related parameters including the channels, number of samples, initial sampling rate, and leaf nodes for which data is to be acquired. All these input parameters are sent to the leaf nodes reliably to set the application. Then, time synchronization is performed to confirm that the leaf nodes' clocks are adjusted. We need to provide reasonable alignment in sensing and permit the tight scheduling of sends using the common Time-Division Multiple Access (TDMA) protocol. TDMA is implemented to allow multiple leaf nodes to communicate with a single receiver, or the sink, by transmitting in different times.

Once synchronization is done, a message for estimating calculating the appropriate delay in sending for the communication protocol is sent to the responsible upstream nodes. We set nodes to send two initialization messages for reliability (successful completion of the application). Additional time is allocated for these messages. Once the system begins sensing, the sensing and sending protocols start working and continue working until the leaf nodes have acquired and sent all the expected number of samples. This centralized computing framework is based on a real-time data acquisition [Linderman et al. 2011], which is particularly developed for the SHM system.

6.2.2. Decentralized Computing/Monitoring. This is the `e-Sampling` scheme that includes both adaptive sampling algorithms and event computation at the node level, that is, a node compares its decision with its neighboring nodes and makes a decentralized decision. The decision from each node then needs to be transmitted to the sink.

Figure 9 illustrates the overall application dataflow of the network under a decentralized computing framework. At the start of the application, the system user inputs the sensing parameters, including the channels, number of samples, initial sampling rate, and sensor nodes, instruction for the event (e.g., damage) indication information sharing and pairwise comparing, and which data is to be sent the sink. These parameters are sent to the sensors reliably in order to initialize the application. After sampling at a required rate and getting samples, each node is required to compute the event detection indication. Each node then shares its results with its other neighboring sensor nodes within its communication range by following a TDMA schedule in which the

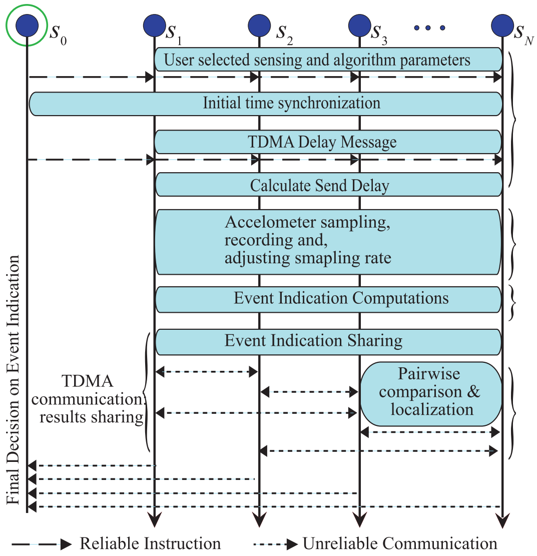

S_0
S_1
S_2
S_3
• • •
S_N
User selected sensing and algorithm parameters
Initial time synchronization
TDMA Delay Message
Calculate Send Delay
Accelometer sampling, recording and, adjusting smapling rate
Event Indication Computations
Event Indication Sharing
Pairwise comparison & localization
TDMA communication results sharing
Final Decision on Event Indication
Reliable Instruction
Unreliable Communication

Fig. 9. Program flow under in the case of decentralized computing.

order of the broadcasts is defined by the ID number of the nodes. When a node receives the strength of the event indication through Equation (14) coming from another node it recomputes the indication in order to locate the event.

If the sink wants to set the current event indication value as the reference one for subsequent tests, then this result is also stored in the external serial flash memory of the node. Otherwise, the current result is matched against the one already stored in the external serial flash memory. At the end of the results sharing phase, the same TDMA schedule is exploited when each node transmits its event indication strength computed as in Equation (15) to the sink node.

It is important to note, to obtain a simplified pairwise event indication strength sharing, a high-precision time synchronization is required to avoid phase differences in the sensors' collected data. We use a modified FTSP (Flooding Time Synchronization Protocol [Maroti et al. 2004]) to fulfill the needs of time synchronization in a hop-by-hop pairwise sharing between sensor nodes, sending the decision toward the upstream nodes and the sink. Instead of flooding time-sync messages to the sensor nodes directly, the sink node multicasts a time-sync message to select downstream nodes using the relevant semantics. Each sensor node is assumed to have a tiny software module or a hardware tool attached to it so that when there is an event locally, the neighboring nodes are enforced to wake up if it is sleeping mode, it failed to detect the event, or the event is not in its sensing area. An example of such a software module is SnoozeAlarm [Rice and Spencer 2009], which is used for SHM applications. It can also be customized for other event monitoring applications.

In `e-Sampling`, we integrate a radio-triggered wake-up module-based synchronization technique. The details of the module can be found in Bhuiyan et al. [2014] and Liu et al. [2013]). Our evaluation setup includes such a module (see Figure 14(a)). When a node has an event indication that is obtained by the signal processing algorithm used in the earlier section, the module is enabled at once to generate interrupt wake-up messages to wake up the corresponding node once it makes a positive decision about an event, for example, "damage," or collects a positive decision from a neighboring node. In other words, a node receives wireless packets sent from others that contain decisions similar to its own, or contains enough signal energy information in the case of raw data transmission.

7. EVALUATION

7.1. Simulation Studies

7.1.1. Methods and System Parameters. We evaluate the applicability and effectiveness of `e-Sampling` in realistic simulations. The focus on conducting the simulations is in the following aspects: (1) sampling performance; (2) computation of event indication; (3) energy cost and energy saving; (4) network lifetime. We use a real dataset that consists of the vibration signals collected from a sophisticated SHM system, deployed on a high-rise GNTVT [Li et al. 2010; Ni et al. 2009]. These datasets are collected by a set of 200 and a set of 800 sensors. Its basic abstraction and the sample sets efficiently provide for sample processing and application-specific format extensions. The vibration signals come from the sensor nodes located at different positions. In the 200-sensor case, the sensors are deployed in a deterministic manner [Li et al. 2010]. We use the datasets for the 100-sensor case in our simulations. The core of our simulation relies on the sample set data structure that contains data collected at a low to high sampling rate. The frequency varying nature of these signals makes them ideal candidates for adaptive sampling. Filter coefficients tabulated in Jairaj and Subbaraman [2010] are the Quadrature Mirror Filters (QMFs) used for the sampling rate selection, and of a length up to 16.

Simulations are performed with Matlab Toolbox using a FEM [Li et al. 2010] of the structure, and within a $50m \times 400m$ rectangular field, taking into account the area of structural environment, for example, a high-rise building, bridge, aircraft, etc. We inject different levels of physical "damage event" information at 10 sensor locations (by modifying input signals randomly in the datasets).

The total energy cost (E_i) of a sensor i is continuously calculated by using Equation (1), summed, and averaged to provide analysis. The hardware constants for the processor and transceiver are from the Intel Xscale PXA271 [Crossbow Technology Inc. 2007], and the ADC is of a Maxim converter defined for multiple sampling rates [25 html]. The Imote2 uses a CC2420 radio chip for wireless communication. This chip adheres to the IEEE 802.15.4 standard and, thus, utilizes the 2.4GHz wireless spectrum. This spectrum includes numerous channels. We model each sensor with six discrete power levels in the interval {−10dBm, 0dBm}, considering the Imote2's power settings, which can be tuned within the IEEE 802.15.4. The study of different routing protocols is out of the scope of this article; for evaluation purposes, we use the Shortest Path (SP) routing model. A detailed description can be found in Olariu and Stojmenovic [2006] and Singh et al. [1998]. The hardware constants and parameters used in the evaluation are set as follows: processor speed is 13MHz; $\beta = 0.83$V; payload size = 32 bytes; $\overline{q[i]q[i+1]} = 30m$; $Q_1 = 0.0027$; $Q_2 = 0.2572$; $\tau = 0.7$; $c = 2.5$; $r \leq 16$.

Comparison: For rigorous comparisons and validating the performance of `e-Sampling`, we implement five schemes as follows. (i) `SPEM` [Li et al. 2010]: this is a SHM scheme validated on the GNTVT and simulated with fixed-rate sampling. (ii) `FloodNet` [Kho et al. 2009]: a decentralized control algorithm for information-based adaptive sampling. (iii) `QAS` [Masoum et al. 2012]: a quality-aware adaptive sampling (an improved version of a sampling algorithm given in Chatterjea and Havinga [2009]). (iv) `e-Sampling`: this proposed scheme. (v) `e-Sampling-C`: this is a *semicentralized* version of `e-Sampling` that excludes the second stage data reduction, that is, sensors transmit the set of F_h data to the sink directly.

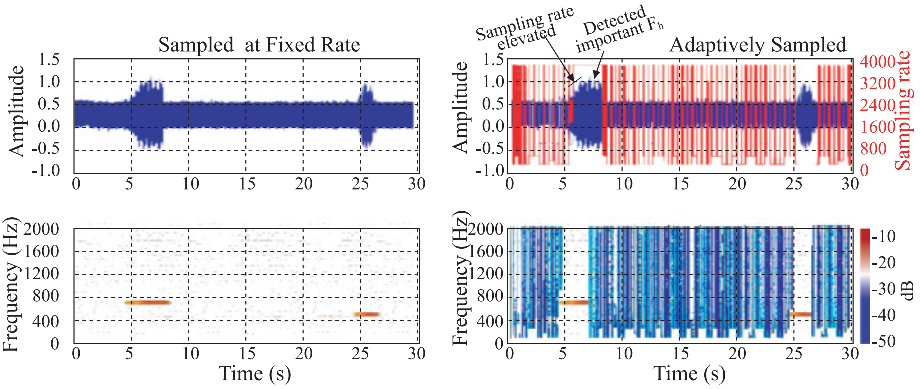


Fig. 10. Adaptive sampling of structural vibration signals.

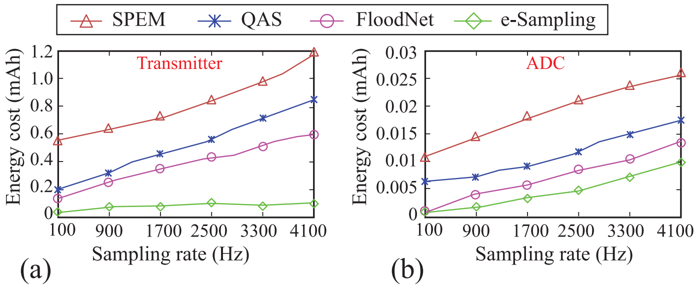


Fig. 11. Energy cost (of (a) transmitter and (b) ADC) of the sixth sensor at different sampling rates calculated over a monitoring round.

7.1.2. Simulation Results. We first study the impact of autonomous adaptive sampling rates of the sensors. Adaptively sampled signals are illustrated in Figure 10, along with the original signals and corresponding F_h. In the right-hand plot, the adjusted sampling rate is plotted in red. Comparing frequencies of the original signals and detected important F_h, it is evident that the adaptive sampling has performed successfully in that the desired signal components can be maintained. The spikes in the sampling rate implies the execution of high-rate sampling, analysis, and sampling rate adjustment. D_l analysis is also adjusted through the algorithm as seen by the difference in time between spikes. Before each set of F_h becomes important, D_h is followed by relatively longer D_l, with a low sampling rate (0–250Hz). When F_h is important, `e-Sampling` calculates D_h and successfully updates the sampling rate. D_l is also adjusted accordingly. It can be best realized by examining the sampling rate plot (right-bottom plot).

Once F_h appears, it is successfully detected by `e-Sampling`. This leads to an increase in the adjusted sampling rate ($R_{h/a}$ is from 2000 to 4100Hz). Thus, D_l shortens. Once F_h has not disappeared (no remarkable changes), D_h shortens and D_l lengthens. This results in a 72.4% decrease in energy cost.

We next study the energy cost and energy saving of different hardware components of a sensor in different schemes, as shown in Figure 11. The energy savings via adaptive sampling for this system can be viewed in Figure 11. By using the datasets with varying sampling rates (between 250 and 4,100Hz), SPEM consumes energy at the rate of about 0.54mAh. It is interesting to mention that when the sampling rate is about 250Hz ($=R_c$) and a burst of sampling rate reaches 4,000Hz ($= R_h$) and gets back again to 250Hz ($R_c \rightarrow R_l$), the energy cost reduces to about 0.07mAh in each interval;

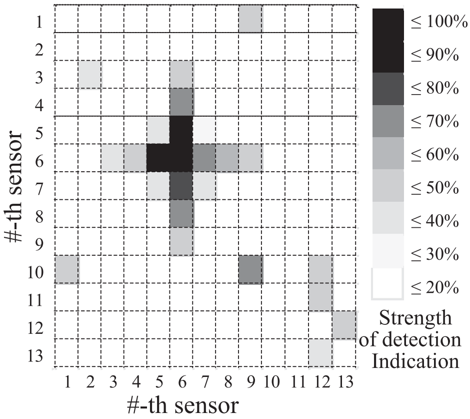


Fig. 12. (a) The strength of indications obtained by the first 13 sensors (comparison between pairs of sensors).

hence, saving about 87%. In the same situation, we found that R_l =1110Hz in QAS and R_l =960Hz in FloodNet. This indicates that the selected minimum sampling rate is still high in these schemes, although there may be no event, that is, the lowest sampling rate cannot be selected in these schemes. The energy cost reduces to about 0.02mAh in FloodNet and about 0.015mAh in QAS, which is very small, compared to e-Sampling. SPEM monitors events at fixed rates, which consume higher energy than both FloodNet and QAS, while e-Sampling outperforms all of the other schemes. The components (e.g., transmitter, ADC, and processing core) in e-Sampling consume less baseline energy than other schemes.

Next, we examine the performance of the event indication in e-Sampling. Figure 12 demonstrates the results under event injection, obtained from the first 13 sensors. In the case of SHM application, since the most dominant *natural frequencies* for physical event detection (e.g., damage) are normally found in part of the F_h, only the frequency range up to 2100Hz is considered in the computation of the event detection indication during the simulation. Here, *natural frequency* is defined by the frequency that a physical structure has a tendency to vibrate at much larger amplitude at some frequencies than others which is different from sampling frequency. We analyze the strength of the event indication between the indication functions $F_{s_u}^{s_v}$ of every pair of sensors.

We see the highest strength of event indication at the sixth sensor location among those sensors. In Figure 12, the strength of the sixth sensor is VHIGH (i.e., more than 81%) and its neighboring sensors also show a high degree of strength. The strengths of the neighbors hints that the presence of an event is highly possible in the vicinity of the sixth sensor. This is why the third, fourth, sixth, and seventh sensors obtain an extent of strength , say, more than 40%. Consequently, they also transmit the indication to the sink, while first, second, and 10th to 13th sensors do not transmit, even if they do not participate in pairwise decision sharing. Thus, damage location can be localized based the comparison of pairs of sensors. We consider the percentage of the detection indication (estimated by the detected changes to the given damage information in the signal). If the strength is more than 40%, a node is required to transmit the indication toward the sink; otherwise, it does not transmit the data.

Since the sixth sensor shows the maximum strengths, next, we verify the performance of e-Sampling in the case of the sixth sensor in different aspects. Figure 13(a) depicts the lifetime of the sixth sensor, regarding the ratio of the D_l and D_h and adaptive sampling rate. The average sampling rate means the average sampling rate used

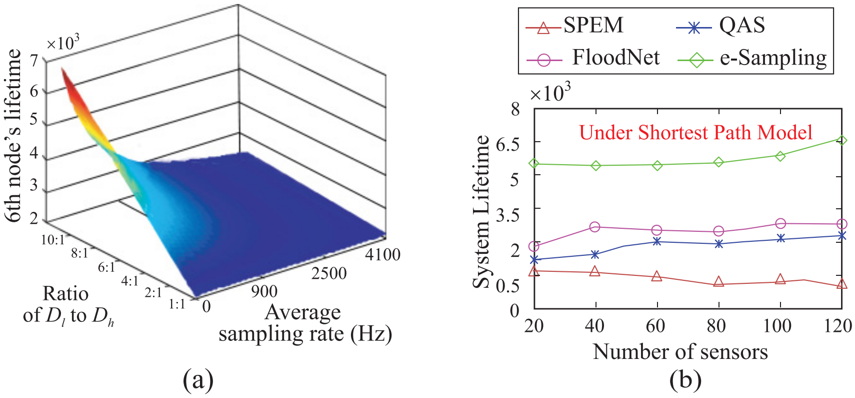


Fig. 13. (a) The lifetime of the sixth sensor vs its adapted sampling rates (with D_l and D_h ratio); (b) the system lifetime.

throughout the entire acquisition; which is specific to the input signal. On the second axis, the ratio of D_l to D_h reflects the amount of time in adjusted sampling rate.

Looking for more details on the network performance, Figure 13(b) depicts that the lifetime increases with the number of sensors. We can see, as expected, e-Sampling performs much better (from 55% to 70%) than the other four schemes. FloodNet slightly outperforms QAS. Their poor performance demonstrates that data collection under event-sensitive sampling and decentralized computing is oblivious to the demand from high-rate applications.

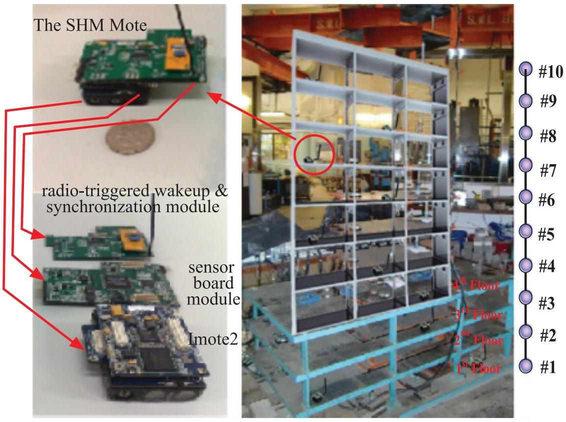
8. PROOF-OF-CONCEPT SYSTEM IMPLEMENTATION

We demonstrate the performance of e-Sampling in real-world settings for both SHM and FEM applications. We describe the evaluation of e-Sampling in FEM application in the Appendix.[1]

8.1. Network Deployment for SHM Applications

We implement a proof-of-concept system using the TinyOS [T2 on Imote2 2010] on Imote2 sensor platforms [Crossbow Technology Inc. 2007]. We specially design a test infrastructure in our laboratory and deploy 10 SHM motes (Integrated Imote2) on it, as shown in Figure 14(a). Three abilities of e-Sampling are justified experimentally, whether or not (i) lowering the sampling rate can lead to a maximized system lifetime, (ii) e-Sampling (the proposed algorithms running on a network) can identify the correct sampling rate in the presence of an event, and (iii) a node is able to automatically and autonomously adjust its sampling rates and intervals.

An additional Imote2 as the sink node is deployed 30 meters away. A PC is used as a command center for the sink and data visualization. Each mote runs a program (implemented in the nesC language) to process the acceleration signal acquired from on-board accelerometers (LIS3L02DQ). The acceleration signal is collected by a digital sensor board at adaptive sampling frequency. The digital acceleration signal is acquired within frames of 4,096 data points and is then stored in the local memory for each round of monitoring. The accelerometer (LIS3L02DQ) has a resolution of 12-bit or equivalent 0.97mg with three-axis of measurement and 2g of amplitude. The three-axis digital accelerometer (LIS3L02DQ) has a +2g measurement range and a resolution of 12 bits or 0.97mg. The LIS3L02DQ is with a built-in ADC, which is followed by digital filters with selectable cutoff frequencies. According to civil engineering, SHM can also be performed by acquiring data within frames of more or less than 560 data points. We



(a) An integrated SHM mote (b) Test building structure (c)

Fig. 14. (a) The SHM mote integrated by Imote2; (b) twelve-story test infrastructure and the placement of 10 SHM motes on it; (c) network topology.

consider 4,096 data points to meet higher requirements posed by diverse network applications.

The Imote2 ADC has digital filters with user-defined cutoff frequencies. Each Imote2 main board combines a low power PXA271 XScale processor with an 802.15.4 radio and an antenna using 2.4GHz. It also offers 256KB of integrated SRAM and 32MB of external SDRAM, although our scheme does not require such a large space. The sink receives the data packets from the sensors (which are decisions on event indication) through wireless communication and relay the data to the PC over a USB cable. The PC commands and sets parameters for the network through sink, analyzes the indication information, and may query the sensors for further information using a Java application. A hardware wake-up with synchronization module is equipped with Imote2, as shown Figure 14(a). This module is connected to Imote2 main board via an external pin of Imote2 (without requiring any extra interface). More details about this radio-triggered wake-up mechanism can be seen in Bhuiyan et al. [2014] and Liu et al. [2013].

Network Deployment on the Designed Structure: A 12-story shear frame structure, made from steel, is employed as shown in Figure 14(b). The lateral stiffness of each floor originates from the four vertical steel columns, 3.81cm × 3.81cm. We instrumented the structure by placing the wireless SHM motes as shown in Figure 14(a). The 10 motes form a network and verify our algorithms for several rounds in several days. The test building has 10 floors; at each floor, each mote is deployed to monitor the structures horizontal accelerations. In the experiment, communication range X_{min} is adjusted by the diameter of the structure. It is easy to adjust since Imote2 supports discrete levels of range.

Physical Event Injection. We inject structural events at the fourth sensor location by removing the plate from the fourth floor of the structure in order to observe the sampling rate adaptation and event indication. The sensor attached on the fourth floor and its neighbors are expected to provide an indication on the presence of the event. To produce a sizable vibration response of the test structure, we collected the original data by vertically exciting the test structure using a magnetic shaker. In fact, this excitation level yields reasonably higher structural responses than a real high-rise structure.

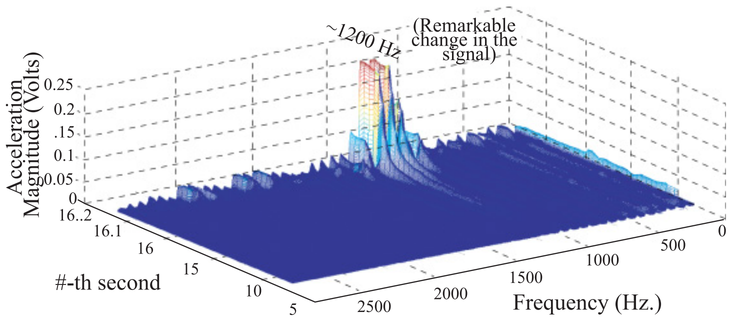


Fig. 15. Waterfall plot of frequency content vs. time.

Table II. Performance of Energy Saving in the WSN-based SHM

Enegy Saving	Deployed sensor no. #									
	1	2	3	4	5	6	7	8	9	10
In the 1st stage	42.3%	27.2%	15.4%	12.1%	21.9%	25.7%	33.8%	62.1%	62.2%	75.1%
In the 2nd stage	53.8%	48.2%	22.5%	13.2%	17.3%	23.5%	43.1%	59%	83.1%	85.2%
Energy Wasting	(-) 4.2% (on average)									

8.1.1. Experiment Results. The performance results of the acquired ambient vibration signals for the case of the fourth sensor are previously shown in Figure 6, in which the important F_h and D_h are analyzed. The results indicate that the original signals and corresponding F_h achieved by our real experiments are quite similar to the results in the simulations illustrated in Figure 10.

When we inject an artificial event on the structure by the hammer strike, `e-Sampling` can successfully identify F_h at the point of time. The sampling rate and corresponding D_l is also adjusted accordingly. This leads to an increase in the adjusted sampling rate (1,100–1,400Hz). For example, this can be best observed by examining the sampling rate waterfall plot in Figure 15. The plot is depicted according to the rate adjusted at a point of time. In most cases, if the high-rate sampling continues, an event is assumed to occur. When the sampling rate is minimum (from 0 to 150Hz), D_l is relatively long. Once the event is injected, that is, there is a change in the F_h, it is detected by our algorithm. This leads to an increase in the adjusted sampling rate (1,100–1,400Hz). The dataset (m) analyzed during D_h is taken to be 4,096 data points. By using (3), the Imote2 at 13MHz completes the task in an estimated 0.2ms.

As with the structural event injection, the sampling rate varies with F_h. The adaptive sampling is performed on all of the 10 sensors' signals, and the results of energy saving caused by data transmission reduction are shown in Table II. Considering the cases like in Figure 15 and Table II, it is evident that the amount of data is significantly reduced, based on the changing bandwidth requirements of the sensors. For the Imote2, a reduction of up to 82% of data is seen at some sensors, in comparison to the 560Hz fixed sampling rate in SPEM. Adaptive sampling enables a net data reduction of 37.4% for the entire acquisition signal, translating to a predicted 34.7% energy saving. This reduction is due to decreased hardware activities in the first stage. This translates to a reduction in the energy cost from 22.6mAh to 17.5mAh. The event indication identified through decentralized computing enables another net data reduction of 44.6% in the second stage, translating to a 41.38% energy saving. About 4.2% energy wasting is estimated from errors, sampling rate selection, and data copy from/to the buffer, and network overhead.

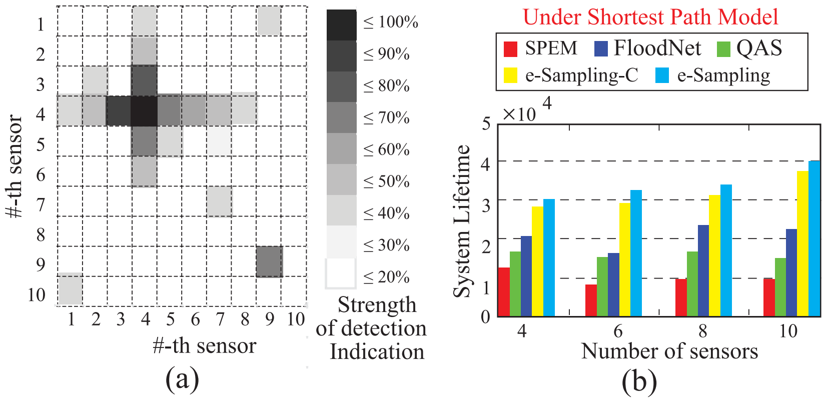


Fig. 16. (a) The strength of event indication by the first 10 sensors; (b) the system lifetime vs. the number of sensors..

Figure 16(a) depicts the strength of the event (i.e., damage) indication at each node, with a comparison between pairs of nodes. For monitoring, the sampling rate is adapted up to 2,500Hz in the experiment. The damage event can be localized based on the comparison of pairs of sensors. The obtained strength of the fourth sensor is VHIGH (i.e., more than 80%) and its neighboring sensors show some degree of strength between MED and HIGH. The sink can verify the event location by analyzing the status of the strength percentage fifth.

Analyzing the results of energy saving in Table II and energy cost, the network lifetime is depicted in Figure 16(b), in which it is evident that this event-sensitive adaptive sampling scheme, `e-Sampling`, has high energy-efficiency in the network, compared to all of the schemes. In SPEM, while collecting data at a fixed rate and transmitting the data in each round, frequent retransmissions for the packet losses decrease energy saving. QAS requires a slightly higher energy cost for all aspects other than FloodNet, except computation.

9. CONCLUSION

We have designed `e-Sampling`, an adaptive data acquisition and low-cost event monitoring scheme for WSNs, as an alternative to the traditional event-insensitive schemes. `e-Sampling` has the ability to automatically switch between high- and low-frequency intervals to reduce the network resource usage and enhance the quality of monitoring. It is quite flexible, as it supports diverse network applications—all the while it is able to run on small and low-power microcontroller-based sensor nodes. Evaluation via simulations and a proof-of-concept system validates the benefits of `e-Sampling`. The results show that, when both algorithms of adaptive sampling and decentralized event indication are used, `e-Sampling` saves up to 87% of the energy consumed by Imote2 sensors.

`e-Sampling` opens a few options for further work. One option can be the performance analysis of the current scheme for monitoring different high-frequency events. As a node-level adaptive sampling for data acquisition, `e-Sampling` is applicable to different high-frequency event monitoring systems that can benefit from prolonged sensor node lifetime. Besides those applications described in the article, more applications include manufacturing, surveillance, and cyberphysical systems such as smart buildings and smart grid. All these applications demand monitoring of systems that reveal signals of fluctuating activity and frequency content, where adaptive sampling can be used to save node energy while retaining signal fidelity. Another option can be the idea of analyzing low- to high-frequency events through graph-based spectral analysis in both space and time domains. Graph Fourier transform and Fourier analysis can be used in this case, where events can be monitored in a sensor network, which can be represented as a graph. Applying such a concept for event detection can be effective compared to `e-Sampling`.

REFERENCES

Application Notes Maxim Corporation, 2-Bit Sampling Analog to Digital Converter Conserves Power 2004. Retrieved from http://www.datasheetarchive.com/AN43-datasheet.html.

F. A. Aderohunmu, D. Brunelli, J. D. Deng, and M. K. Purvis. 2015. A data acquisition protocol for a reactive wireless sensor network monitoring application. *Sensor* 15, 2015 (April 2015), 10221–10254.

M. Z. A. Bhuiyan, G. Wang, J. Cao, and J. Wu. 2015a. Deploying wireless sensor networks with fault-tolerance for structural health monitoring. *IEEE Transactions on Computers* 64, 2 (2015), 382–395.

C. Alippi, G. Anastasi, M. Francesco, and M. Roveri. 2010. An adaptive sampling algorithm for effective energy management in wireless sensor networks with energy-hungry sensors. *IEEE Transactions on Instrumentation and Measurement* 59, 2 (2010), 335–344.

M. Z. A. Bhuiyan, G. Wang, J. Cao, and J. Wu. 2013. Energy and bandwidth-efficient wireless sensor networks for monitoring high-frequency events. In *Proceedings of the 10th IEEE International Conference on Sensing, Communication, and Networking (SECON'13)*. 194–202.

M. Z. A. Bhuiyan, G. Wang, J. Cao, and J. Wu. 2014. Sensor placement with multiple objectives for structural health monitoring. *ACM Transactions on Sensor Networks* 10, 4 (2014), 1–45.

M. Z. A. Bhuiyan, G. Wang, and A. V. Vasilakos. 2015b. Local area prediction-based mobile target tracking in wireless sensor networks. *IEEE Transactions on Computers* 64, 2 (2015), 1968–1982.

M. Z. A. Bhuiyan, G. Wang, J. Wu, J. Cao, X. Liu, and T. Wang. 2015c. Dependable structural health monitoring using wireless sensor networks. *IEEE Transactions on Dependable and Secure Computing* (http://dx.doi.org/10.1109/TDSC.2015.2469655), (2015), 1–14.

M. Z. A. Bhuiyan, J. Wu, G. Wang, and J. Cao. 2016. Sensing and decision-making in cyber-physical systems: The case of structural health monitoring. *IEEE Transactions on Industrial Informatics* (2016), 1–11.

S. Burer and J. Andlee. 2007. Solving maximum-entropy sampling problems using factored masks. *Mathematical Programming* 109, 2 (2007), 263–281.

S. Chatterjea and P. J. M. Havinga. 2009. An adaptive and autonomous sensor sampling frequency control scheme for energy-efficient data acquisition in wireless sensor networks. In *Proceedings of IEEE DCOSS*. 60–78.

Z. Chen, G. Barrenetxea, and M. Vetterli. 2012. Share risk and energy: Sampling and communication strategies for multi-camera wireless monitoring. In *Proceedings of INFOCOM*. 1862–1870.

A. Ciancio, P. Sundeep, O. Antonio, and K. Bhaskar. 2006. Energy-efficient data representation and routing for wireless sensor networks based on a distributed wavelet compression algorithm. In *Proceedings of IEEE IPSN*. 309–316.

Crossbow Technology Inc. 2007. Imote2 hardware reference manual.

D. Donoho and I. Johnstone. 1994. Ideal spatial adaptation by wavelet shrinkage. *Biometrica* 81, 3 (1994), 425–455.

B. Greenstein, C. Mar, A. Pesterev, S. Farshchi, E. Kohler, J. Judy, and D. Estrin. 2006. Capturing high-frequency phenomena using a bandwidth-limited sensor network. In *Proceedings of ACM SenSys*. 1–14.

V. Gutnik and A. Chandrakasan. 1997. Embedded power supply for low-power DSP. *IEEE Transactions on VLSI System* 12, 4 (1997), 425–434.

G. Hackmann, W. Guo, G. Yan, Z. Sun, C. Lu, and S. Dyke. 2014. Cyber-physical codesign of distributed structural health monitoring with wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems* 24, 1 (2014), 63–72.

J. Hao, B. Zhang, Z. Jiao, and S. Mao. 2015. Adaptive compressive sensing based sample scheduling mechanism for wireless sensor networks. *Pervasive and Mobile Computing* 22, C (Sept. 2015), 113–125.

M. Jairaj and S. Subbaraman. 2010. QMF implementation using Xilinx SysGen (XSG). In *Proceedings of IEEE ICCSIT*. 142–135.

F. Keinert. 2004. *Wavelets and Multiwavelets* (1st ed.). Chapman & Hall, New York.

W. Khan, Y. Xiang, M. Aalsalem, and Q. Arshad. 2013. Mobile phone sensing systems: A survey. *IEEE Communications Surveys and Tutorials* 15, 1 (2013), 402–427.

J. Kho, A. Rogers, and N. Jennings. 2009. Decentralized control of adaptive sampling in wireless sensor networks. *ACM Transactions on Sensor Networks* 5, 3 (2009), 1–35.

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

B. Li, D. Wang, and Y. Q. Ni, F. Wang. 2010. High quality sensor placement for SHM systems: Refocusing on application demands. In *Proceedings of INFOCOM*. 1–9.

L. E. Linderman, K. A. Mechitov, and B. F. Spencer. 2011. *Real-Time Wireless Data Acquisition for Structural Health Monitoring and Control*. Technical Report NSEL-029. University of Illinois at Urbana-Champaign, IL.

X. Liu, J. Cao, S. Lai, C. Yang, H. Wu, and Y. Xu. 2011. Energy efficient clustering for WSN-based structural health monitoring. In *Proceedings of INFOCOM*. 2768–2776.

X. Liu, J. Cao, and S. Tang. 2013. Enabling fast and reliable network-wide event-triggered wakeup in WSNs. In *Proceedings of the IEEE 24th Real-Time Systems Symposium (RTSS'13)*.

H. Luo, J. Wang, Y. Sun, H. Ma, and X.-Y. Li. 2010. Adaptive sampling and diversity reception in multi-hop wireless audio sensor networks. In *Proceedings of IEEE ICDCS*. 378–387.

A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson. 2002. Wireless sensor networks for habitat monitoring. In *Proceedings of ACM WSNA*. 88–97.

M. Maroti, B. Kusy, G. Simon, and A. Ledeczi. 2004. The flooding time synchronization protocol. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys'04)*. 39–49.

A. Masoum, N. Meratnia, and P. J. M. Havinga. 2012. A decentralized quality aware adaptive sampling strategy in wireless sensor networks. In *Proceedings of UIC/ATC*. 298–305.

S. Mitra. 2005. *Digital Signal Processing: A Computer Based Approach* (3rd ed.). McGraw-Hill College.

T. Nagayama, B. F. Spencer, G. A. Agha, and K. A. Mechitov. 2006. Model-based data aggregation for structural monitoring employing smart sensors. In *Proceedings of INSS*.

Y. Ni, Y. Xia, W. Liao, and J. Ko. 2009. Technology innovation in developing the structural health monitoring system for Guangzhou New TV Tower. *Structural Control and Health Monitoring* 16, 1 (2009), 73–98.

S. Olariu and I. Stojmenovic. 2006. Design guidelines for maximizing lifetime and avoiding energy holes in sensor networks with uniform distribution and uniform reporting. In *Proceedings of INFOCOM*. 1–9.

J. A. Rice and B. F. Spencer. 2009. *Flexible Smart Sensor Framework for Autonomous Full-Scale Structural Health Monitoring*. Technical Report NSEL. Report.018. Newmark Structural Engineering Laboratory, University of Illinois at Urbana-Champaign, Champaign, Illinois.

N. Saito and R. Coifman. 1995. Local discriminant bases and their applications. *Journal of Mathematical Imaging and Vision* 5, 4 (1995), 337–358.

W. Shu, X. Liu, Z. Gu, and S. Gopalakrishnan. 2008. Optimal sampling rate assignment with dynamic route selection for real-time wireless sensor networks. In *Proceedings of IEEE RTSS*. 431–441.

D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst. 2013. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *Signal Processing Magazine* 30, 3 (2013), 83–98.

S. Singh, M. Woo, and C. Raghavendra. 1998. Power-aware routing in mobile ad hoc networks. In *Proceedings of ACM MobiCom*. 180–191.

I. Sobron, P. Diniz, W. A. Martins, and M. Velez. 2015. Energy detection technique for adaptive spectrum sensing. *IEEE Transactions on Communications* 63, 3 (2015), 617–627.

Texas Instruments. 2013. ADC12DL065: Analog-to-digital converter technology. Retrieved from http://www.ti.com/product/ADC12DL065.

T2 on Imote2. 2010. Homepage. Retrieved from http://tinyos.stanford.edu/tinyos-wiki/index.php/T2_on_Imote2.

N. Tsiftes and A. Dunkels. 2011. A database in every sensor. In *Proceedings of ACM SenSys*. 316–332.

G. Wang, M. Z. A. Bhuiyan, and Z. Li. 2010a. Two-level cooperative and energy-efficient tracking algorithm in wireless sensor networks. *Concurrency and Computation: Practice & Experience* 22, 4 (2010), 518–537.

J. Wang, Y. Liu, and S. K. Das. 2010b. Energy-efficient data gathering in wireless sensor networks with asynchronous sampling. *ACM Transactions on Sensor Networks* 6, 3 (2010), 1–44.

R. Willet, A. Martin, and R. Nowak. 2004. Backcasting: Adaptive sampling for sensor networks. In *Proceedings of ACM/IEEE IPSN*. 124–133.

L. Xu, X. Hao, N. D. Lane, X. Liu, and T. Moscibroda. 2015. Cost-aware compressive sensing for networked sensing systems. In *Proceedings of IEEE IPSN*. 130–141.

S. Yun and L. Qiu. 2015. Supporting Wi-Fi and LTE co-existence. In *Proceedings of INFOCOM 2015*. 1–9.

Received December 2015; revised July 2016; accepted August 2016