Fidelity-Aware Utilization Control for Cyber-Physical Surveillance Systems

Jinzhu Chen, Rui Tan, Guoliang Xing, *Member*, *IEEE*, Xiaorui Wang, *Member*, *IEEE*, and Xing Fu, *Student Member*, *IEEE*

Abstract—Recent years have seen the growing deployments of Cyber-Physical Systems (CPSs) in many mission-critical applications such as security, civil infrastructure, and transportation. These applications often impose stringent requirements on system *sensing fidelity* and *timeliness*. However, existing approaches treat these two concerns in isolation and hence are not suitable for CPSs where system fidelity and timeliness are dependent on each other because of the tight integration of computational and physical resources. In this paper, we propose a holistic approach called *Fidelity-Aware Utilization Controller* (FAUC) for Wireless Cyber-physical Surveillance (WCS) systems that combine low-end sensors with cameras for large-scale ad hoc surveillance in unplanned environments. By integrating data fusion with feedback control, FAUC can enforce a CPU utilization upper bound to ensure the system's real-time schedulability although CPU workloads vary significantly at runtime because of stochastic detection results. At the same time, FAUC optimizes system fidelity and adjusts the control objective of CPU utilization adaptively in the presence of variations of target/noise characteristics. We have implemented FAUC on a small-scale WCS testbed consisting of TelosB/Iris motes and cameras. Moreover, we conduct extensive simulations based on real acoustic data traces collected in a vehicle surveillance experiment. The testbed experiments and the trace-driven simulations show that FAUC can achieve robust fidelity and real-time guarantees in dynamic environments.

Index Terms—Real-time detection, data fusion, CPU utilization control, cyber-physical systems.

1 Introduction

Cyber-physical System (CPS) is a new class of embedded systems that tightly integrate computational and physical resources. Recent years have seen the growing deployments of CPSs in many mission-critical applications such as security, civil infrastructure, and transportation. These applications often impose stringent performance requirements including *sensing fidelity* and *timeliness*. In this work, we define fidelity as a system's capability of reaching correct conclusions even when the sensing results from the dynamic physical environment are noisy. In addition to fidelity, timeliness is another fundamental requirement as many computational tasks in a CPS must complete within tight deadlines in order to avoid undesirable or even catastrophic consequences.

In this work, we investigate the problem of addressing both fidelity and timeliness requirements of Wireless Cyber-physical Surveillance (WCS) systems. A typical WCS system consists of battery-powered cameras, sensors, and embedded computers that communicate through wireless networks. Without the reliance on wired power/communication infrastructure, WCS systems can be rapidly deployed in an ad hoc manner for large-scale surveillance in *unplanned* environments. This is a key advantage for many critical domains such as security, transportation, and natural/physical hazard monitoring. In 2008, a number of wirelessly connected cameras were deployed for real time and high-fidelity surveillance over a 26-mile course of the Boston Marathon which attracted over 20,000 runners and more than one million spectators [1]. In other scenarios like border security, WCS systems need to provide surveillance and intruder detection during an extended period of time up to several years. Because of the tight budget on power resources and network bandwidth, WCS systems often operate in an on-demand fashion where low-end (e.g., acoustic/infrared/magnetic) sensors serve as "sentinels" that wake up high quality but power consuming sensors (e.g., pan-tilt-zoom cameras) once a possible target is detected. High-quality sensing results (e.g, images) are then transmitted to an embedded computer for high-fidelity object detection and recognition.

Both fidelity and timeliness are essential requirements of the WCS systems described above. As an example, users may require any target of interest to be detected "at high fidelity (both missing and false alarm rates lower than 1 percent) and in real time (delay within 5 seconds)." However, a key challenge is that the timeliness and fidelity of a WCS system are tightly dependent on each other. First, the performance of low-end sensors is extremely sensitive to dynamics in the physical environment. It is shown in [2] that individual dual-axis magnetometers on Mica2 motes [3] can exhibit up to 60 percent false alarm and missing rates. As low-end sensors trigger image capture and

- J. Chen, R. Tan, and G. Xing are with Department of Computer Science and Engineering, Michigan State University, 3115 Engineering Building, East Lansing, MI 48824. E-mail: {chenjinz, tanrui, glxing}@msu.edu.
- X. Wang is with Department of Electrical and Computer Engineering, Ohio State University, 508 Dreese Laboratories, 2015 Neil Avenue, Columbus, OH 43210. E-mail: xwang@ece.osu.edu.
- X. Fu is with Department of Electrical Engineering and Computer Science, The University of Tennessee, 800 Longview Rd, APT204, Knoxville, TN 37996. E-mail: xfu1@utk.edu.

Manuscript received 16 Sept. 2011; accepted 20 Dec. 2011; published online 17 Feb. 2012.
Recommended for acceptance by S. Papavassiliou, N. Kato, Y. Liu, C.-Z. Xu, and X. Wang.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDSSI-2011-09-0667.
Digital Object Identifier no. 10.1109/TPDS.2012.74.

processing, their poor fidelity can significantly affect the workload and real-time performance of the system. For instance, the false alarms from low-end sensors not only lead to energy waste of cameras but also generate extra computation workload for image processing. On the other hand, reducing CPU workload and camera activity unnecessarily may lead to the increased target missing rate.

In this paper, we argue that the fidelity and real-time concerns of WCS systems must be *jointly* addressed because of the tight integration of system computational and physical components. Numerous real-time scheduling algorithms have been proposed to achieve real-time guarantees for computing systems. However, many of them require detailed knowledge of CPU workload while WCS systems are subject to stochastic workload because of the impact of physical dynamics. Several recent approaches [4], [5], [6] can handle variable system workload. However, they are incognizant of system fidelity requirements. On the other hand, although sensor calibration [7] and signal processing [8] techniques are available to improve the fidelity of a sensing system, they do not account for the impact on system timeliness. For instance, minimizing target missing rate often leads to a high false alarm rate [8], which in turn poses undesirable CPU workload for a WCS system as discussed earlier.

In this work, we propose a novel approach to holistically addressing the fidelity and timeliness requirements of WCS systems. Our approach integrates multisensor data fusion [8] with feedback control to achieve adaptive fidelity and real-time guarantees for WCS systems operating in dynamic environments. Specifically, we make the following major contributions in this paper:

1. We propose a novel problem formulation for the fidelity-aware utilization control problem where a given upper bound on the CPU utilization is enforced while system detection error rate is minimized. Our formulation is based on two rigorous performance models that characterize the fusion-based detection performance and the expected CPU utilization induced by processing stochastic detection results.
2. We develop Fidelity-Aware Utilization Controller (FAUC) that adaptively adjusts the data fusion threshold to bound the CPU utilization according to user requirement. At the same time, FAUC minimizes the system detection error rate while ensuring real-time schedulability.
3. We have implemented FAUC on a small-scale WCS testbed consisting of TelosB motes, Iris motes, and cameras. Our extensive experiments on light and acoustic target detection show that FAUC can achieve robust fidelity and utilization control in the presence of significant physical dynamics and unreliable wireless links.
4. We conduct extensive simulations based on real acoustic data traces collected in a military vehicle surveillance experiment. The simulations evaluate the performance of FAUC under a wide range of settings of network size and signal-to-noise ratio.

The rest of the paper is organized as follows: Section 2 reviews related work. Section 3 presents the background on sensing and data fusion models. Section 4 describes our problem and provides an overview of our approach. Sections 5 and 6 model system performance and present the design of FAUC, respectively. Sections 7 and 8 present the testbed experiment results and trace-driven simulation results, respectively. Section 9 concludes the paper.

2 Related Work

Data fusion [8] is an effective signal processing technique that improves the fidelity of sensing systems by mitigating the impact of noise. Most previous studies [8] focus on analyzing the optimal fusion strategy of a given sensing system. In our earlier work [9], [10], we study the impact of data fusion on spatial and temporal coverage of large-scale sensor networks. Sensor calibration can also improve system fidelity by correcting sensor biases. In [11], the biases of light sensors are estimated by solving the equations that correlate their measurements. Similarly, in [7], the parameters of ranging sensors are estimated based on pairwise range measurements. In [12], sensors are jointly calibrated to improve the system-level performance of fusion-based sensor networks. The above approaches calibrate sensors according to known ground-truth inputs and hence work in an open-loop fashion. In our recent work [12], we develop a feedback-based calibration algorithm that maintains system sensing fidelity in the presence of environmental dynamics. However, data fusion and sensor calibration are not concerned with meeting timing constraints.

Feedback control techniques have shown great promise in providing real-time guarantees for CPSs by adapting to workload variations based on dynamic feedback. For instance, feedback-based CPU utilization control [4], [5], [6] has been demonstrated to be an effective way of meeting the end-to-end deadlines for real-time systems. However, most of these algorithms rely on task rate adaptation and hence cannot handle unpredictable task rate variations that may be caused by low system fidelity. Different from these studies, we aim to jointly address the requirements on system fidelity and CPU utilization of WCS systems.

3 Preliminaries

In this section, we present the preliminaries of our work, which include sensor measurement and data fusion models.

3.1 Sensor Measurement Model

We assume that sensors measure the energy of received signals for event detection. Let s_i denote the signal energy received by sensor i, which is affected by several factors and varies for different sensors. First, each sensor may have its hardware bias. Second, the measurement value is stochastic as it inevitably contains environmental noise. Third, the signal path loss between the event and sensor varies with distance and terrain. Depending on the hypothesis that the target is absent (H_0) or present (H_1), the measurement of sensor i, denoted by y_i, is given by

$$H_0 : y_i = n_i, \qquad H_1 : y_i = s_i + n_i,$$

where n_i is the energy of noise in sensor i's measurement. We assume that the noises of sensors are independent and follow the normal distributions, i.e., $n_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$, where μ_i and σ_i^2 are the mean and variance of n_i, respectively. The sensor measurement model described above has been widely adopted in the literature of event detection [13] and also has been empirically verified [14], [15]. However, many previous studies assume that the parameters of the above model, i.e., s_i, μ_i, and σ_i, are known a priori. Unfortunately, this assumption often does not hold in reality because of the stochastic nature of sensing. In this paper, we assume that these parameters are unknown.

3.2 Data Fusion Model

Data fusion [8] has been proposed as an effective signal processing technique to improve the system performance of sensing systems. A system based on data fusion is usually organized into multiple clusters. Each cluster has a cluster head that gathers information from member sensors and makes the system decision regarding the presence of the target. We adopted a simple data fusion model where the system decision is made by comparing the sum of member sensors' measurements against a threshold T, which is referred to as the *fusion threshold* hereafter. The cluster head makes a positive decision if the sum of measurements exceeds the threshold. Such a model has been adopted by several previous studies [9], [12]. Suppose there are N sensors in a cluster. The sum of measurements, denoted by Y, is given by $Y = \sum_{i=1}^{N} y_i$. Let $\widetilde{H}_0$ and $\widetilde{H}_1$ represent the detection decisions that the target is absent and present, respectively. Denote $S = \sum_{i=1}^{N} s_i$, $\mu = \sum_{i=1}^{N} \mu_i$ and $\sigma^2 = \sum_{i=1}^{N} \sigma_i^2$. Depending on whether the target is present, the sum of sensor measurements follows the normal distribution, i.e., $Y|H_0 \sim \mathcal{N}(\mu, \sigma^2)$ or $Y|H_1 = \sum_{i=1}^{N} \sim \mathcal{N}(\mu + S, \sigma^2)$. A target is detected only if the sum of sensor measurements is greater than the threshold T, i.e., $Y > T$. The detection of a target is inherently stochastic because of the random noises in sensor measurements. The *system* detection performance is characterized by two metrics, namely, the false alarm rate (denoted by P_F) and missing probability (denoted by P_M). P_F is the probability of deciding $\widetilde{H}_1$ when *no* target is present, and P_M is the probability of deciding $\widetilde{H}_0$ when a target is present. P_F and P_M can be computed by $P_F = Q(\frac{T-\mu}{\sigma})$ and $P_M = Q(-\frac{T-\mu-S}{\sigma})$, where $Q(x)$ is the Q-function of the standard normal distribution, i.e., $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^{+\infty} \exp(-\frac{t^2}{2}) \mathrm{d}t$.

4 Problem Statement

In this section, we describe the problem of fidelity-aware utilization control. We first discuss the system model in Section 4.1. We then formulate our problem and provide a brief overview of our approach in Section 4.2.

4.1 System Model

We assume that a Wireless Cyber-physical Surveillance system consists of a base station and multiple sensor clusters. Each cluster is composed of *low-end* and *high-quality* sensors. The low-end sensors (e.g., acoustic and infrared sensors) usually have a low manufacturing cost and low energy consumption. As a result, their sensing capability is often limited. As we discussed in Section 3.2, the system performance can be improved by employing data fusion on the measurements of low-end sensors. The high-quality sensors (e.g., pan-tilt-zoom cameras [16] and active radars [17]) can provide high-accuracy sensing and detection at the price of higher manufacturing cost and energy consumption. In this paper, we assume that there is only one high-quality sensor in each cluster. However, our approach can be easily extended to the case of multiple high-quality sensors, where they may fuse their measurements to yield a detection result.

In accordance with the heterogeneous system architecture, we adopt a *two-phase* target detection process. Initially, the low-end sensors periodically sense the environment while the cameras remain asleep because their power consumption is typically several orders of magnitude higher than low-end sensors [3]. The cluster head fuses data from low-end sensors and makes a decision according to the threshold T. If it is a positive decision, the cluster head then activates the camera to capture an image of the surveillance region, and sends the image to the base station. Finally, a target recognition algorithm is executed by the base station to process the images and detect whether a target of interest is present. The key advantage of such a two-phase target detection scheme is that the system power consumption can be significantly reduced without sacrificing the detection performance. In particular, most false alarms can be filtered out by the data fusion of low-end sensors and hence the high-quality sensors (i.e., cameras) can sleep for most of the time and be switched on only when the probability of target presence is high. As a result, the system can achieve high-fidelity surveillance for extended lifetime in unplanned environments without wired power infrastructure. We note that, our approach is not restricted to the particular WCS system architecture described above. It can be applied to similar heterogeneous system architectures where computation intensive tasks are triggered by the sensing results of low-power sensors.

4.2 Problem Formulation

Our objective is to achieve satisfactory timeliness and fidelity of WCS systems. We now describe the formulation of our problem and provide a brief overview of our approach.

To guarantee the end-to-end timeliness required by a WCS system, the delay of each stage of the entire process of sensing, communication, and computing must be carefully considered. In previous studies [18], achieving real-time sensor sampling, data fusion, and wireless communications in surveillance systems has been extensively studied. In this paper, we focus on providing the real-time guarantee on target detection in base station that must run computation-intensive tasks to process high-quality sensor data such as images. Specifically, we control the CPU utilization to enforce appropriate schedulable utilization bounds, e.g., the Liu and Layland bound for rate-monotonic scheduling [19], despite significant uncertainties in system workloads. In the meantime, utilization control can also enhance system survivability by providing overload protection against workload fluctuation [20]. Our approach can be integrated with previous solutions [18] to ensure the end-to-end timeliness of a WCS system. For instance, the deadline of target detection can be ensured by enforcing subdeadlines for sensing, communication, and computing separately. We note that the delay of computation is often significant when complex sensing data such images need to be processed.

Several challenges must be addressed to satisfy both timeliness and fidelity requirements simultaneously for WCS systems. First, the timeliness and fidelity performance of a system are highly dependent on each other. For instance, although the false alarms of low-end sensors can be dealt with by turning on the camera more frequently, it inevitably increases CPU workload and impedes system timeliness. On the other hand, reducing CPU workload and camera activity

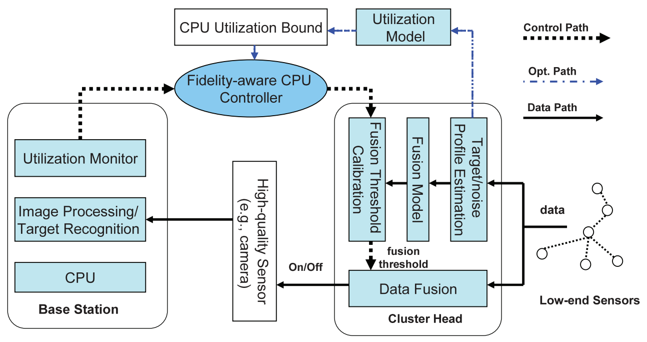


Fig. 1. The architecture of FAUC controller.

aggressively may lead to an increased target missing rate. Second, system CPU workloads are highly variable because of several factors such as uncertain image processing time and stochastic detection performance of low-end sensors. The probability that the camera is activated and an image needs to be processed is highly dependent on the data fusion results, which in turn are affected by time-varying noise and target characteristics in dynamic environments.

In this paper, we propose a control-theoretic solution called Fidelity-Aware Utilization Controller to address these challenges. FAUC employs a feedback controller to enforce the specified upper bound on CPU utilization of base station while minimizing the overall system detection error rate. By taking advantage of the adaptivity of the controller, FAUC allows a WCS system to achieve robust assurance on timeliness and fidelity in dynamic environments. We formally formulate our problem as follows:

Problem 1 (Fidelity-Aware Utilization Control). *To find a stable and converging control algorithm for the fusion threshold T at the cluster head based on the feedback of the base station, such that the expected CPU utilization $\mathbb{E}[u]$ is upper bounded by u_s while the detection error rate P_e is minimized, where u_s is a constant that ensures system's real-time schedulability.*

In the above formulation, the CPU utilization bound u_s is a predefined user input to the controller, and we focus on the utilization control of only one cluster. When there exist multiple clusters in the system, their CPU utilization bounds can be determined by schedulability analysis [19] and then separately enforced by multiple FAUC controllers. The detection error rate P_e in Problem 1 is the probability that the system makes a wrong detection decision, which jointly accounts for false alarms and misses. Such a metric is widely adopted in the literature of sensing systems [8]. We choose fusion threshold T as the control input as it affects both the system detection performance and timeliness. Specifically, when T is lower, the missing rate is lower while more false alarms may be triggered by noise leading to higher system workload. On the other hand, a higher T reduces both false alarm rate and system workload while a target is more likely to be missed.

Fig. 1 illustrates the architecture of FAUC and a WCS system. We now describe the three main components in the system architecture. 1) *Two-phase fusion-based target detection* (Section 5). The measurements of low-end sensors are first fused by the cluster head. If a positive decision is made, the camera is activated and the captured image is then processed by the base station for target recognition. 2) *Utilization feedback control loop* (Sections 6.1 and 6.2). The FAUC utilization controller adaptively calibrates the fusion threshold T to enforce a user-specified utilization upper bound when the system workloads vary significantly as a result of stochastic camera activations and uncertain image processing time. The utilization monitor measures the average CPU utilization and provides feedback for calibrating the fusion threshold. The controller design is based on analytical models of CPU utilization and system fidelity. 3) *Detection performance optimization loop* (Sections 6.3 and 6.4). FAUC employs the k-means clustering algorithm [21] to periodically estimate system parameters (e.g., target and noise models) from sensor measurements. The results are used to optimize the fusion threshold and adjust the control objective of CPU utilization. In the presence of physical dynamics (e.g., variations of target/noise energy), such an optimization mechanism adapts utilization control objectives and maintains satisfactory system fidelity.

5 Performance Modeling

In this section, we formally model the performance of WCS systems. The results provide a foundation for the design of FAUC controller in Section 6. We first model the system detection error rate in Section 5.1. The impact of communication packet loss is analyzed in Section 5.2. Finally, we model system CPU utilization in Section 5.3.

5.1 System Detection Performance

Before formally modeling the system detection performance, we first make the following assumptions. First, the probability that a target is present at any time instance is P_a, which is unknown but can be estimated from detection history. Second, the false alarm rate and missing probability of the high-quality sensor, denoted by P_{FH} and P_{MH}, are known. P_{FH} and P_{MH} can often be measured via offline experiments. Because of the high accuracy of the high-quality sensor, both P_{FH} and P_{MH} are close to zero. In addition, we let P_{FL} and P_{ML} denote the false alarm rate and missing probability of low-end sensors. The system detection error rate P_e is the weighted sum of the joint false alarm rate and missing probability of high-quality and low-end sensors. Specifically,

$$P_e = (1-P_a)\cdot P_{FL}\cdot P_{FH} + P_a\cdot (P_{ML} + (1-P_{ML})\cdot P_{MH}), \qquad (1)$$

where $P_{FL}\cdot P_{FH}$ corresponds to the case that both the low-end sensors and the camera raise a false alarm, and $P_{ML} + (1-P_{ML})\cdot P_{MH}$ corresponds to the case that the low-end sensors make a correct detection but the activated camera misses the event. We now discuss how to achieve the minimal P_e. In Section 3.2, we have derived the expressions for the false alarm rate and missing probability of low-end sensors, i.e., $P_{FL} = Q(\frac{T-\mu}{\sigma})$ and $P_{ML} = Q(-\frac{T-\mu-S}{\sigma})$. By replacing P_{FL} and P_{ML} in (1) with these expressions and solving the condition for minimal P_e, i.e., $\frac{\partial P_e}{\partial T} = 0$, the optimal fusion threshold T_{opt} that minimizes P_e is given by

$$T_{\mathrm{opt}} = \frac{\delta\cdot\sigma^2}{2S} + \mu + \frac{S}{2},$$

where $\delta = 2\ln(\frac{1-P_a}{P_a}\cdot\frac{P_{FH}}{1-P_{MH}})$. Note that the performance modeling above is based on the assumption that all the packets sent by low-end sensors are correctly received.

5.2 Impact of Packet Loss

Packet loss due to unreliable wireless communication can cause the system detection performance of low-end sensors to deviate from the theoretical results derived in Section 5.1. We propose to address the impact of packet loss by exploiting *temporal sampling*, where the number of samples each low-end sensor transmits to the cluster head is determined by the quality of communication links. In this section, the quality of a communication link from a sensor to the cluster head is characterized by the end-to-end packet reception rate. Suppose sensor i samples m times in a detection process. Let y_{ij} denote the j^{th} noisy measurement of sensor i in a detection, p_i denote the end-to-end packet reception rate of the path from sensor i to the cluster head, and $u_{ij} \in \{0,1\}$ denote the packet delivery state of measurement y_{ij}. Hence, u_{ij} is a Bernoulli random variable with success probability of p_i. In the temporal sampling scheme, the cluster head fuses all measurement received during a detection process to make a decision, where the fusion statistic Y is given by $Y = \sum_{i=1}^{N}\sum_{j=1}^{m} u_{ij}\cdot y_{ij}$. In the absence of target, the mean and variance of Y are given by

$$\begin{aligned}\mathbb{E}[Y|H_0] &= \sum_{i=1}^{N}\sum_{j=1}^{m}\mathbb{E}[u_{ij}\cdot y_{ij}|H_0]\\ &= \sum_{i=1}^{N}\sum_{j=1}^{m}\mathbb{E}[u_{ij}]\cdot\mathbb{E}[y_{ij}|H_0] = m\cdot\sum_{i=1}^{N}p_i\cdot\mu_i,\end{aligned}$$

$$\begin{aligned}\mathrm{Var}[Y|H_0] &= \sum_{i=1}^{N}\sum_{j=1}^{m}\mathrm{Var}[u_{ij}\cdot y_{ij}|H_0]\\ &= \sum_{i=1}^{N}\sum_{j=1}^{m}\mathbb{E}\big[u_{ij}^2\cdot y_{ij}^2|H_0\big] - \big(\mathbb{E}[u_{ij}\cdot y_{ij}|H_0]\big)^2\\ &= \sum_{i=1}^{N}\sum_{j=1}^{m}\big(\mathbb{E}\big[u_{ij}^2\big]\cdot\mathbb{E}\big[y_{ij}^2|H_0\big] - p_i^2\cdot\mu_i^2\big)\\ &= m\cdot\sum_{i=1}^{N}p_i\cdot\sigma_i^2 + \mu_i^2\cdot(p_i - p_i^2).\end{aligned}$$

Note that $\mathbb{E}[\mu_{ij}^2] = p_i$ and $\mathbb{E}[y_{ij}^2|H_0] = \sigma_i^2 + \mu_i^2$. Similarly, in the presence of target, the mean and variance of $Y|H_1$ are given by

$$\mathbb{E}[Y|H_1] = m\cdot\sum_{i=1}^{N}p_i\cdot(s_i+\mu_i), \tag{2}$$

$$\begin{aligned}\mathrm{Var}[Y|H_1] &= m\cdot\sum_{i=1}^{N}p_i\cdot\big(\sigma_i^2 + (s_i+\mu_i)^2\big) - p_i^2\cdot(s_i+\mu_i)^2\\ &= m\cdot\sum_{i=1}^{N}p_i\cdot\sigma_i^2 + (s_i+\mu_i)^2\cdot(p_i-p_i^2).\end{aligned} \tag{3}$$

Note that $\mathbb{E}[y_{ij}^2|H_1] = \sigma_i^2 + (s_i+\mu_i)^2$. As Y is the sum of $N\cdot m$ independent random variables, according to the Central Limit Theorem (CLT), Y follows the normal distribution when $N\cdot m$ is large enough. Although N is often limited in practice, we can increase the number of samples during a detection to satisfy the condition of CLT. Denote $\mu' = m\cdot\sum_{i=1}^{N}p_i\cdot\mu_i$, $S' = m\cdot\sum_{i=1}^{N}p_i\cdot s_i$, $\sigma'_{H_0} = \sqrt{\mathrm{Var}[Y|H_0]}$ and $\sigma'_{H_1} = \sqrt{\mathrm{Var}[Y|H_1]}$. The system false alarm rate and missing probability of low-end sensors are given by $P_{FL} = Q(\frac{T-\mu'}{\sigma_{H_0}})$ and $P_{ML} = Q(-\frac{T-\mu'-S'}{\sigma_{H_1}})$, respectively. Therefore, when the impact of packet loss cannot be ignored, we need to separately estimate the variances for the cases of target absence and presence, respectively. By replacing P_{FL} and P_{ML} in (1), the equation $\frac{\partial P_e}{\partial T} = 0$ has two roots, which are

$$\frac{\sigma_{H_0}^2 S' - \sigma_{H_1}^2\mu' + \sigma_{H_0}^2\mu' \pm \sigma_{H_1}\sigma_{H_0}\sqrt{\delta'\sigma_{H_1}^2 - \delta'\sigma_{H_0}^2 + S'^2}}{\sigma_{H_0}^2 - \sigma_{H_1}^2},$$

where $\delta' = 2\ln\frac{(1-P_a)P_{FH}\sigma_{H_1}}{P_a(1-P_{MH})\sigma_{H_0}}$.[1] The optimal fusion threshold T_{opt} that minimizes P_e is one of the above roots that gives the smaller P_e.

5.3 System CPU Utilization

To guarantee the real-time schedulability (e.g., by rate-monotonic scheduling [19]), the CPU utilization of each task at the base station shall be maintained at a certain level. In this section, we derive the CPU utilization model. The CPU workload of the base station is mainly generated by processing the images captured by the camera. As the camera is activated by the stochastic decisions of data fusion, the CPU workload is hence subject to change over time. We define that a *control cycle* consists of m *detections*. In each detection, low-end sensors send their measurements to the cluster head for data fusion. We now derive the expected CPU utilization in m detections of a control cycle, denoted by $\mathbb{E}[u]$, by accounting for the workload generated by both correct decisions and false alarms.

We define the following notations subject to a control cycle: 1) n_{f1} and n_{d1} are the numbers of false alarms and correct detections made by the cluster head, respectively. Note that they are unknown to the system, 2) n_{f2} and n_{d2} are the numbers of positive decisions made by the cluster head but regarded as false alarms and correct detections by the camera, respectively. These two numbers can be counted by the base station after processing the images of the camera. We have the following relationships:

$$n_{f1} + n_{d1} = n_{f2} + n_{d2}, \tag{4}$$

$$n_{f2} = n_{f1}\cdot(1-P_{FH}) + n_{d1}\cdot P_{MH}, \tag{5}$$

$$n_{d2} = n_{f1}\cdot P_{FH} + n_{d1}\cdot(1-P_{MH}). \tag{6}$$

Equation (4) holds because either a correct decision or false alarm from data fusion would trigger an image processing task at the base station. The result of image processing can then again be classified as correct decision or false alarm. In (5), $n_{f1}\cdot(1-P_{FH})$ represents the number of false alarms that are correctly identified by the high-quality sensor, and $n_{d1}\cdot P_{MH}$ represents the number of correct detections that are incorrectly decided as false alarms. In (6), $n_{f1}\cdot P_{FH}$ represents the number of false alarms that are incorrectly decided as correct detections, and $n_{d1}\cdot(1-P_{MH})$ represents the number of detections that are correctly identified. From (5) and (6), the unknown n_{f1} and n_{d1} can be estimated as

1. We assume $\delta'\sigma_{H_1}^2 - \delta'\sigma_{H_0}^2 + S'^2 \geq 0$ such that there exists optimal fusion threshold minimizing P_e; otherwise, P_e will be a monotonically decreasing function of T and there is no optimal fusion threshold.

$$n_{f1} = \frac{n_{f2}(1 - P_{MH}) - n_{d2}P_{MH}}{1 - P_{FH} - P_{MH}},$$
$$n_{d1} = \frac{n_{d2}(1 - P_{FH}) - n_{f2}P_{FH}}{1 - P_{FH} - P_{MH}}.$$

Therefore, the estimates of P_{FL} and P_{ML}, denoted by $\widetilde{P}_{FL}$ and $\widetilde{P}_{ML}$, respectively, are given by

$$\widetilde{P}_{FL} = \frac{n_{f1}}{m - m \cdot P_a}, \quad \widetilde{P}_{ML} = \frac{m \cdot P_a - n_{d1}}{m \cdot P_a}. \tag{7}$$

The high-quality sensor sends the data to the base station for image processing, which consumes the CPU resource. We assume that the average CPU execution time with or without processing an image is e or e', respectively. Note that e' may equal zero if no processing is required when the data fusion produces a negative result. Let T_d represent the duration of a control cycle and u represent the CPU utilization of the base station. The *expected* CPU utilization, denoted by $\mathbb{E}[u]$, is given by

$$\mathbb{E}[u] = (n_{f1} + n_{d1}) \cdot \frac{e}{T_d} + (m - n_{f1} - n_{d1}) \cdot \frac{e'}{T_d}. \tag{8}$$

By replacing n_{f1} and n_{d1} in (8) with (7), we have

$$\begin{aligned}\mathbb{E}[u] &= \frac{m \cdot e'}{T_d} + \frac{m \cdot (e - e')}{T_d}((1 - P_a) \cdot \widetilde{P}_{FL} + P_a \cdot (1 - \widetilde{P}_{ML})) \\ &\simeq K_1 + K_2\left((1 - P_a)Q\left(\frac{T - \mu}{\sigma}\right) + P_a Q\left(\frac{T - \mu - S}{\sigma}\right)\right),\end{aligned} \tag{9}$$

where $K_1 = \frac{m \cdot e'}{T_d}$ and $K_2 = \frac{m \cdot (e - e')}{T_d}$. When the impact of packet loss cannot be ignored, $\widetilde{P}_{FL}$ and $\widetilde{P}_{ML}$ should be replaced by $P_{FL} = Q(\frac{T - \mu'}{\sigma_{H_0}})$ and $P_{ML} = Q(-\frac{T - \mu' - S'}{\sigma_{H_1}})$, respectively, where S', μ', σ_{H_0} and σ_{H_1} are derived in Section 5.2. From (9), the expected CPU utilization monotonically decreases with T because both P_{FL} and $1 - P_{ML}$ decrease with T.

6 Fidelity-Aware CPU Controller

Based on the performance modeling presented in Section 5, we first design the fidelity-aware CPU controller in Section 6.1 and discuss its stability and convergency in Section 6.2. After that, we discuss the estimation of system parameters in Section 6.3 and the approach to optimizing the detection error rate in Section 6.4.

6.1 The Design of FAUC

The objective of Problem 1 is to ensure $\mathbb{E}[u] \leq u_s$ while minimizing system detection error rate P_e, where $\mathbb{E}[u]$ is a function of the fusion threshold T given by (9) and u_s is a user-specified utilization bound. As the threshold T is calibrated every control cycle, Problem 1 is a typical discrete-time control problem, in which u_s is the *reference*, T is *control input* and $\mathbb{E}[u]$ is the *controlled variable*. In the following, we present the design of Fidelity-Aware Utilization Controller. We first discuss how FAUC ensures the utilization bound, i.e., $\mathbb{E}[u] \leq u_s$. In Section 6.4, we discuss how to minimize the system detection error rate P_e under the given utilization bound.

The block diagram of the FAUC feedback control loop is shown in Fig. 2. The key challenge of deriving the transfer function $G_p(z)$ is that $Q(x)$ in (9) is a nonlinear function. In this paper, we adopt the linear approximation of (9), which is given by

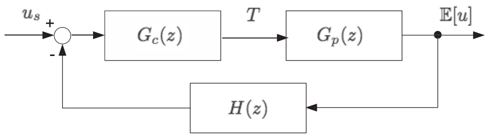


Fig. 2. The closed-loop system to control the fusion threshold according to the CPU utilization feedback.

$$\mathbb{E}[u](T) \simeq \mathbb{E}[u](T_0) + \left.\frac{\partial \mathbb{E}[u]}{\partial T}\right|_{T_0} \cdot (T - T_0),$$

where the derivative $\frac{\partial \mathbb{E}[u]}{\partial T}$ is given by

$$\frac{\partial \mathbb{E}[u]}{\partial T} = -\frac{K_2}{\sigma\sqrt{2\pi}}\left((1 - P_a) \cdot \mathrm{e}^{-\frac{(T-\mu)^2}{2\sigma^2}} + P_a \cdot \mathrm{e}^{-\frac{(T-\mu-S)^2}{2\sigma^2}}\right). \tag{10}$$

T_0 is referred to as the *operating point*, which greatly affects the control performance. We will discuss how to choose T_0 in Section 6.3. Hereafter, we denote $F(T_0) = \frac{\partial \mathbb{E}[u]}{\partial T}|_{T_0}$. By taking z-transform to the linear approximation, we have $G_p(z) = F(T_0)$. Therefore, the system to be controlled is a zero-order system. We can estimate $\mathbb{E}[u]$ based on the samples of CPU utilization at the base station in a control cycle. This estimate is then fed back to compare with the reference u_s. As the estimation takes one control cycle, $H(z) = z^{-1}$ which represents a delay of one cycle. We now design $G_c(z)$ to solve Problem 1. As $G_p(z)$ is zero order, a first-order controller is sufficient to meet stability and convergence requirements. Therefore, we let $G_c(z)$ be

$$G_c(z) = \frac{a}{1 - b \cdot z^{-1}}, \tag{11}$$

where $a > 0$ and $b > 0$. The coefficients a and b should be chosen to ensure the system stability and convergence. The conditions of system stability and convergence are analyzed in Section 6.2.

6.2 Stability and Convergence

We first analyze the system stability. The closed-loop transfer function, denoted by $T_c(z)$, is given by $T_c(z) = \frac{G_c(z)G_p(z)}{1+G_c(z)G_p(z)H(z)} = \frac{a \cdot F(T_0) \cdot z}{z-(b-a \cdot F(T_0))}$. The closed-loop system has a pole at $z = b - a \cdot F(T_0)$. From control theory [22], if the pole is within the unit circle centered at the origin, i.e., $|b - a \cdot F(T_0)| < 1$, the system is stable. As $F(T_0)$ is always negative, the sufficient condition for stability is $\frac{b+1}{F(T_0)} < a < \frac{b-1}{F(T_0)}$.

We then analyze the steady-state error of the system. The open-loop transfer function, denoted by $T_o(z)$, is given by $T_o(z) = G_c(z)G_p(z)H(z) = \frac{a \cdot F(T_0)}{z-b}$. By letting $b = 1$, the system is a type I system [22], in which the controlled variable $\mathbb{E}[u]$ can converge to the reference u_s provided that the system is stable. Hence, by replacing b with 1, the condition for both stability and convergence is $\frac{2}{F(T_0)} < a < 0$.

According to Fig. 2, we have $G_c(z) = \frac{T(z)}{u_s - H(z)\mathbb{E}[u](z)}$. By replacing $G_c(z)$ with (11) and letting $H(z) = z^{-1}$, we have the z-domain equation $T(z) = b \cdot T(z) \cdot z^{-1} + a \cdot (u_s - z^{-1} \cdot \mathbb{E}[u](z))$, which is equivalent to the equation in time-domain as $T[n] = b \cdot T[n-1] + a \cdot (u_s - \mathbb{E}[u][n-1])$, where $\mathbb{E}[u][n-1]$ is the average CPU utilization measured in the $(n-1)^{\text{th}}$ calibration cycle, $T[n]$ and $T[n-1]$ are the fusion thresholds for the n^{th} and $(n-1)^{\text{th}}$ control cycle, respectively.

6.3 Online System Parameter Estimation

There exists a fundamental tradeoff between the stability and transient response performance of a control system [22]. In our problem, the system converges faster for a larger a at the price of greater system oscillations. Therefore, the best setting for a is a value close to its lower bound $\frac{2}{F(T_0)}$. The stability can be enforced if we can online estimate the $F(T_0)$. We now discuss how to choose T_0 and estimate $F(T_0)$. According to (10), in order to compute $F(T_0)$, several parameters, i.e., e, e', μ, σ, and S, need to be estimated. Because e and e' are subject to change because of the stochastic task execution time, we calculate the average execution time for the tasks for the estimation. Assuming that μ, σ, and S can slowly change over time because of the uncertainty of the environment, we employ the k-means [21] clustering algorithm to estimate these parameters. Specifically, the k-means algorithm iteratively constructs two clusters of sensor measurements which correspond to the cases of target absence and presence, respectively. The noise mean μ is estimated by averaging the measurements in the cluster representing the noise. The target energy S can be calculated by subtracting μ from the average of measurements in the cluster representing the case of target presence. Note that if the impact of packet loss can be ignored, the variance σ^2 is estimated by averaging the variances from two clusters; otherwise, we use the separate variances from the two clusters. As the CPU utilization shall be controlled around u_s, the operating point T_0 of the linearization is obtained by solving $\mathbb{E}[u] = u_s$ where $\mathbb{E}[u]$ is given by (9) with the estimated μ, σ^2, and S. With the chosen T_0, we can compute $F(T_0)$.

6.4 Optimizing Detection Error Rate

So far, our discussion is only concerned with controlling the utilization bound while the impact on detection error rate is not considered. Although such a solution can meet the deadline once a target is detected, it may lead to low system fidelity such as excessive false alarms and consume unnecessary energy. In this section, we discuss how to optimize system detection performance without violating the utilization bound.

According to our performance modeling in Section 5.1, the detection performance is optimized if the fusion threshold T is set to T_{opt}. However, we cannot simply adjust the current threshold to T_{opt} without accounting for the impact on CPU utilization. FAUC addresses this issue by implementing a *dual cycle* control strategy. In each *control* cycle, CPU utilization is enforced to the utilization bound u that is initially set to the user-specified constant u_s. Each *optimization* cycle consists of multiple control cycles. The system parameters are estimated every optimization cycle as discussed in Section 6.3 and then used to update T_{opt} and compute the expected utilization u^* according to our utilization model (9). If the estimated utilization is lower than the initial utilization bound, i.e., $u^* < u_s$, it will be set as the new control objective for the following control cycles until the start of next optimization cycle. Therefore, the optimization process opportunistically lowers the utilization bound if the system detection performance can be optimized. In other words, the CPU utilization never exceeds the initial value specified by user. Hence, the real-time schedulability is always satisfied.

7 Testbed Experiments

To evaluate the performance of FAUC, we have conducted two testbed experiments for detecting light and acoustic targets, respectively. The results allow us to evaluate the effectiveness of our approach for different sensor modalities. Section 7.1 discusses the experimental methodology. Sections 7.2 and 7.3 present the experimental results of detecting light and acoustic targets, respectively. Sections 7.4 and 7.5 evaluate the impact of packet loss and number of sensor nodes to the system by using the light spot detection experiment. Section 7.6 shows a multicluster system that works with FAUC to have all the clusters' utilization well controlled.

7.1 Experimental Methodology

We adopt a baseline approach referred to as *fixed-step closed-loop heuristic* for performance comparison. In this heuristic algorithm, the expected CPU utilization $\mathbb{E}[u]$ is fed back to compare with the reference u_s. As $\mathbb{E}[u]$ is a decreasing function of threshold T, if $\mathbb{E}[u] > u_s$, the new threshold $T[n]$ is calculated by adding a fixed step Δ_T to the previous threshold $T[n-1]$; otherwise, Δ_T is subtracted from previous $T[n-1]$ to calculate new threshold $T[n]$ if $\mathbb{E}[u] < u_s$. However, this approach does not consider system stability and convergence. In the experiment, we employ different Δ_T for this approach to evaluate the impact of step size on the system performance.

Our evaluation focuses on two performance metrics: utilization error and average detection error rate. The utilization error is the error between the measured CPU utilization and the reference in each control cycle. In order to calculate the detection error rate, we log the ground-truth information regarding the target appearance to compute the system false alarm and missing rates in each control cycle. The detection error rate is then computed as the weighted sum of the false alarm and missing rates, with weights $(1 - P_a)$ and P_a, respectively.

7.2 Light Spot Detection

In the light spot detection experiment, four TelosB motes [3] and a webcam are attached against the LCD screen of a desktop computer to detect a light spot that is randomly displayed on the screen (see Fig. 3). The display of the light spot is controlled by a computer program, simulating the random presence of the target at a probability of $P_a = 50\%$ in each 1 second time slot. Note that a similar method has been adopted by previous work [12]. We vary the light intensity of each pixel on the LCD screen to simulate the changeable characteristics of noise and target. To create noise in sensor measurements, each pixel is assigned a small random light intensity value I_N with the mean of μ. I_N varies over time to simulate the changing environmental noise. To create the target, a constant light intensity value I_T

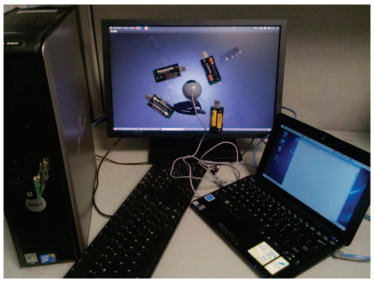


Fig. 3. Testbed for light spot detection. Four TelosB motes and a webcam detect the light spot.

is added to the noise for each pixel. Note that I_T decreases with the distance from the center of the light spot. Similarly, I_T varies to simulate the change of target profile. The TelosB motes measure the light intensity every 250 milliseconds via the on-board Hamamatsu S1087-01 light sensors [3] and transmit the measurements to the cluster head that is connected to a base station laptop. The cluster head fuses the readings received within every 250 milliseconds and detects the light spot. When the cluster head makes a positive decision, the webcam is triggered to take an image and compare the average intensity over all pixels with a threshold. The false alarm and missing rates of the webcam are 5.1 and 3.9 percent, respectively, which are estimated in a separate offline experiment. We evaluate the utilization control algorithms under a variety of settings. Moreover, each experiment consists of two phases. Specifically, in the first phase, the mean of I_N as well as I_T remain at their initial values. The second phase starts after 12 control cycles, where we vary the mean of I_N or I_T to simulate the changes of noise or target profile.

7.2.1 Sensor Measurement Performance

We first verify the Gaussian noise assumption made in Section 3. Fig. 5 plots the Cumulative Distribution Function (CDF) of a light sensor's measurements in office environment. We can see from the figure that the sensor measurements well match the normal distribution. We then evaluate the performance of the k-means algorithm that is used to estimate the online noise and target profiles. Note that the FAUC adjusts the control reference based on the estimates given by the k-means algorithm. Therefore, the estimation accuracy can affect the performance of FAUC. Fig. 6 plots the CDF of relative errors of the estimated μ, σ^2, and S with respect to their ground truths, respectively. We can see that all of them can be estimated accurately. For instance, about 40 percent of the estimated σ^2 has a near-zero error and the maximum error is only about 2 percent.

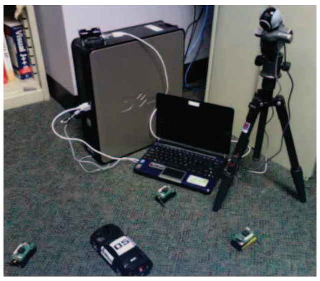


Fig. 4. Testbed for acoustic target detection. Three Iris motes and a webcam detect the moving toy car.

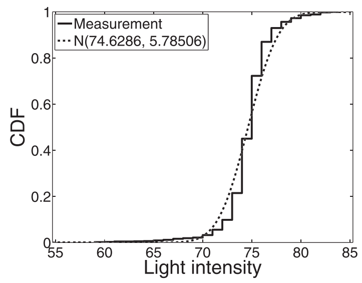


Fig. 5. The CDF of the light intensity measurements of a TelosB mote.

7.2.2 Stability and Convergence

We now evaluate the stability and convergence of FAUC in dynamic environments. Fig. 9 shows the temporal evolution of the system when noise mean is increased at the 12$^{\text{th}}$ control cycle. Each optimization cycle comprises five control cycles. The initial CPU utilization reference is set to be 0.62. Based on this setting, FAUC computes an initial fusion threshold T, which is very low as shown in the top sub figure in Fig. 9. As a result, the noise occasionally exceeds the fusion threshold causing a false alarm rate of about 5 percent. At the end of the first optimization cycle, i.e., the 5th control cycle, FAUC computes the optimal fusion threshold T_{opt} based on the estimated target/noise parameters. As $T_{\text{opt}} > T$, there exists an opportunity to reduce system false alarms. FAUC thus computes a new utilization bound of 0.38 based on T_{opt}, which causes the controller to increase T. The bottom two sub figures in Fig. 9 show that the measured utilization quickly converges to the new reference and system error rate drops to zero.

When the noise energy is increased at the 12th control cycle, the CPU utilization increases accordingly because false alarms from low-end sensors trigger extra image processing tasks. FAUC then attempts to lower the utilization by increasing T. At the next optimization cycle, i.e., the 15th control cycle, FAUC estimates the target/noise parameters and computes a new T_{opt} that is lower than T. As the utilization reference (about 0.6) that corresponds to the new T_{opt} is still lower than the initial bound 0.62, FAUC increases the reference to reduce false alarms, as discussed in Section 6.4. At the next optimization cycle, i.e., the 20th control cycle, T_{opt} exceeds the current T. FAUC then lowers the utilization reference again, which frees more CPU resources. The above results demonstrate several salient features of FAUC when it operates in dynamic environments. First, it yields satisfactory

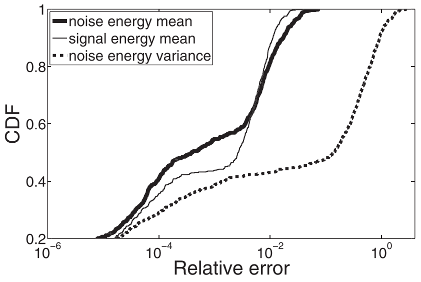


Fig. 6. The CDF of the estimation errors of noise and target profiles.

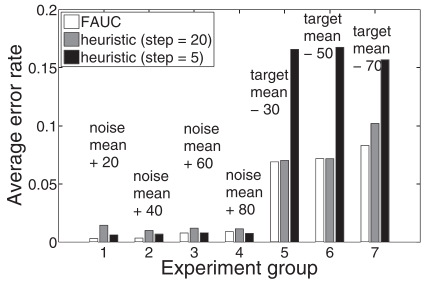


Fig. 7. The average detection error rate under different target/noise dynamics.

control performance as the CPU utilization quickly converges to the reference even when false alarms introduce unpredictable system workloads. Second, FAUC can effectively adapts the utilization reference to minimize system error rate.

7.2.3 *Effectiveness*

We now compare the performances of various approaches in seven groups of experiments with a variety of target and noise dynamics. In the first four groups of experiments, we increase the noise mean from 20 to 80 with a step of 20. In the last three groups of experiments, we decrease the target signal by 30, 50, and 70, respectively. Fig. 7 shows the average detection error rate in different group of experiments. We can see that all three approaches can maintain small detection error rates when the noise increases. Although the increasing noise causes more false alarms for low-end sensors, the false alarms can be effectively filtered out by the image-based target detection. When the target signal decreases, all three approaches yield higher error rates. In particular, the heuristic algorithm with a step size of 5 misses about 16 percent targets. When the step size is 20, it has fewer misses because the controller settles faster. FAUC achieves the minimal error rate under all settings.

Fig. 8 plots the CDF of the utilization errors in the seven groups of experiments. We can see that FAUC significantly outperforms the heuristic algorithm with various settings. In particular, about 80 percent of errors of FAUC fall within 10 percent. In contrast, the heuristic algorithm does not exploit the relationship between the control input and the controlled variable and hence yields considerable steady-state errors. We can conclude from Figs. 7 and 8 that FAUC yields excellent control performance while maintaining satisfactory fidelity even when target/noise have dynamic characteristics.

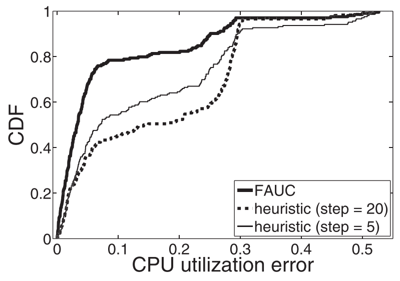


Fig. 8. The CDF of the absolute CPU utilization error.

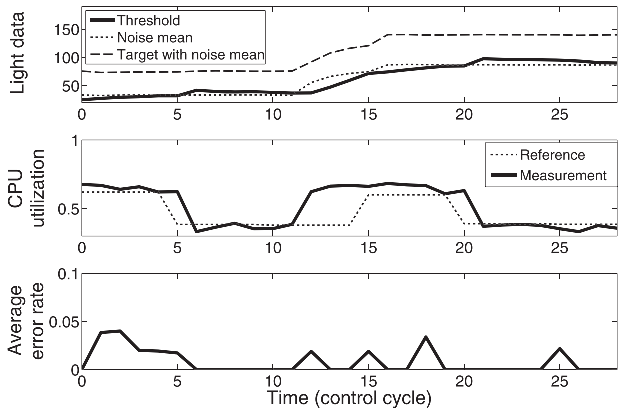


Fig. 9. The temporal evolution of the light spot detection in dynamic environments.

7.3 Acoustic Target Detection

In the second set of testbed experiments, we use three Iris motes [3] and a webcam to detect a radio-controlled toy car. Fig. 4 illustrates the setup of the experiments. The cluster head is connected to the base station laptop. The microphone of MTS300 sensor board [3] on Iris mote samples at 100 Hz to detect acoustic signals from the toy car. Each mote calculates the acoustic signal energy every 50 samples and transmits to the cluster head every 500 milliseconds. The cluster head fuses the measurements received from the Iris motes and activates the webcam to capture an image if a positive decision is made. The base station compares the image with the stored background image to detect the target using the ImageMagick tool [23]. Note that another webcam connected to another computer records the ground-truth information. It is challenging to detect the toy car using the low-cost Iris motes because of the significantly dynamical characteristics of the toy car. Specifically, as the toy car moves through the surveillance region quickly, the acoustic signal emitted by the car varies significantly with the car's location and speed.

Fig. 10 shows the evolution of system performance over 16 control cycles. The CPU utilization reference is set to be 0.3. We intentionally adopt such a low reference to study the tradeoff between utilization and detection performance. Each optimization cycle comprises four control cycles and

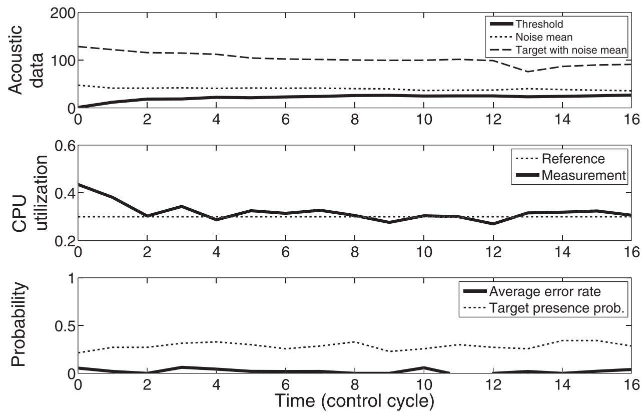


Fig. 10. The temporal evolution of the acoustic target detection in dynamic environments.

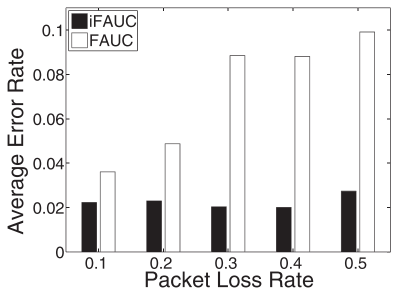


Fig. 11. Average detection error under different packet loss rate.

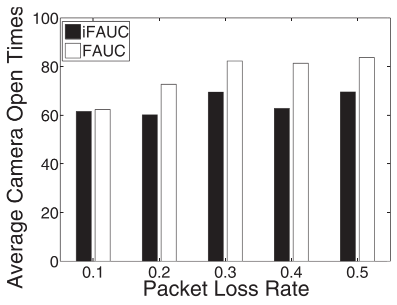


Fig. 12. Average camera switched-on times under different packet loss rate.

each control cycle comprises 70 detections. The toy car moves along a circular path that crosses the surveillance region. The target appearance probability varies during the experiment as shown in the bottom sub figure of Fig. 10. The fusion threshold is initially set to a low value, which leads to many false alarms and image processing tasks. This in turn causes higher CPU utilization which exceeds the required CPU utilization reference. In response, FAUC increases the fusion threshold and hence the CPU utilization quickly drops to the reference at the second control cycle. Afterward, the CPU utilization is well controlled at the reference. The slightly changing target appearance probability leads to slightly changing CPU utilization and fusion threshold. In addition, the car's acoustic signal happens to decrease at the 13th cycle, which causes a higher false alarm rate and may increase the CPU utilization. However, the target appearance probability is slightly reduced at that time so that the CPU utilization does not increase substantially. To adapt to this change, the threshold also slightly decreases. Overall, the results of this experiment show that the system is robust to the variations of target energy and appearance probability.

7.4 Impact of Packet Loss

As shown by the analysis of impact of packet loss on FAUC in Section 5.2, we can improve the performance by increasing the number of sensor nodes or number of samplings per detection in the packet loss. Based on this result, we design an improved FAUC (iFAUC) and evaluate it in light detection experiment. In FAUC, each sensor takes one sample every 250 milliseconds and sends to the cluster head for data fusion. On the contrary, each sensor in iFAUC samples every 25 milliseconds and transmits to the cluster head once every 10 samples. Taking each sample as a packet, we apply different packet loss rates to iFAUC and FAUC for evaluation. The experiments are conducted in the same way as described in the Section 7.2. Fig. 11 shows the average detection error rate comparison between iFAUC and FAUC under the different packet loss rates. Although both methods achieve low average detection error rate, iFAUC significantly outperforms FAUC under all the packet loss rates. Due to the better accuracy, iFAUC has more chances to reduce the CPU utilization reference, thus releasing more CPU resources. Comparison of camera switched-on times between iFAUC and FAUC shows iFAUC always switch the camera on less frequently than FAUC in all packet loss rates (Fig. 12). As the camera switched-on times is proportional to the CPU utilization in (8), iFAUC outperforms FAUC again regarding average CPU utilization.

7.5 Impact of the Number of Sensor Nodes

In CPSs that adopt sensor data fusion, the number of sensors may affect the performance of the system. Based on the light detection experiment, we conducted experiments with different number of sensors to evaluate the resulted performance. We vary the number of sensor from one to four and conduct experiment described in Section 7.2. From Fig. 13, the average error rate doubles when there is only one sensor compared to two sensors. We can see from Fig. 13 that the error rate of four sensors is slightly higher than that of three sensors. In the experiment, the fourth sensor receives the lowest signal-to-noise ratio (SNR). As the data fusion rule adopted in this work treats all sensors with equal weight, fusing the reading from a sensor with low SNR may not be beneficial. We note that if we adopt more effective data fusion rule that accounts for sensors' SNRs [8], system detection performance would not degrade even if low-SNR readings are fused.

7.6 Multicluster CPSs

We now extend our evaluation to the case of multiple clusters that may exist for large CPSs. In order to assure the real-time performance of the base station whose resources are shared by all clusters, the CPU utilization of the base station shall be carefully maintained according to the real-time scheduling strategy. In this experiment, we deploy two clusters, each of which is used to monitor a light spot. Two light spots are generated by computer program, separately, with a target present probability of 50 percent. One high-quality camera is used for each cluster to confirm the presence of the light spots, respectively. However, we deploy different number of sensor nodes and apply the different control and estimation cycle length for the two clusters. Cluster 1 contains two sensor nodes and comprises 100 samples per control cycle with 400 samples per estimation cycle. On the other hand, the cluster 2 contains

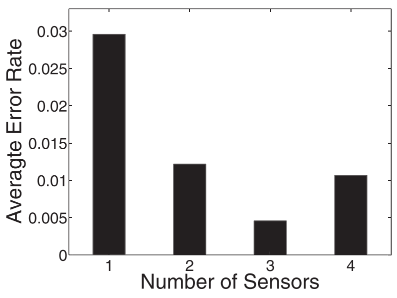


Fig. 13. Average error rate under different number of sensor nodes.

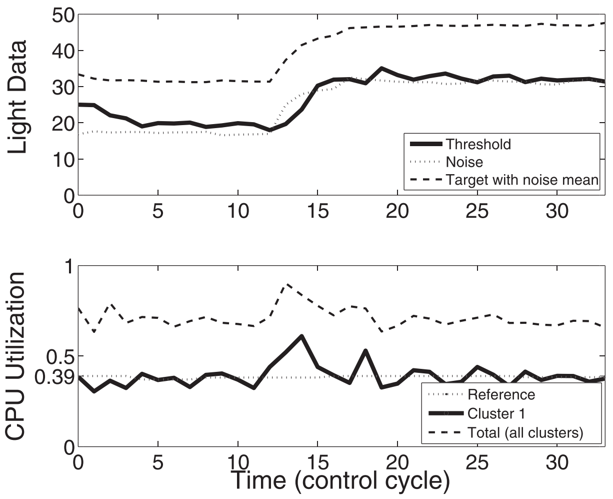


Fig. 14. The temporal evolution of the light target detection in dynamic environments of cluster 1. CPU utilization reference is set to 39 percent.

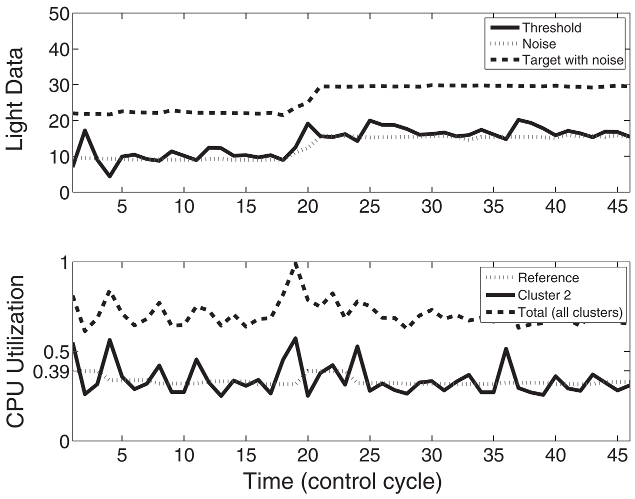


Fig. 15. The temporal evolution of the light target detection in dynamic environments of cluster 2. CPU utilization reference is set to 39 percent.

only one sensor nodes and comprises 80 samples per control cycle with 320 samples per estimation cycle. We conduct the experiment with changeable noise for both light spots. Based on the real-time scheduling strategy, we set the base station CPU utilization's upper bound to be 78 percent and then assign 39 percent utilization for each cluster.

Fig. 14 shows the time evolution for cluster 1 in light spot detection. It performs similarly as in the previous light spot detection experiment. In the first 12 control cycles, CPU utilization is controlled around 39 percent. We can see the system tries to reduce the CPU utilization reference for cluster 1 at the fourth cycle. At the 12th cycle, the noise increases, hence the CPU utilization consumed by cluster 1 increases due to the higher false alarm rate of low-end sensors. This change on the CPU utilization of cluster 1 is also reflected in the total CPU utilization. Then the system quickly adapts the threshold to reduce the CPU utilization of cluster 1 to the set point within three cycles. Accordingly the total CPU utilization of base station reduces to a lower level than the reference 78 percent. At the 16th cycle, the estimated T_{opt} requires higher CPU utilization. However this higher utilization is still lower than the original user setpiont for cluster 1. As a result, the system increases the CPU utilization reference accordingly. The system then controls the CPU utilization of cluster 1 on this reference for the rest of time. Similarly, Fig. 15 shows the time evolution for the cluster 2. As there is only one sensor node in the cluster, it is more likely to have higher false alarm rate and missing probability. However, from the 5th, 20th, 25th, and 37th cycle, we can see the system adapts to the CPU utilization changes very quickly by adjusting the threshold under the environment dynamics. From the time evolutions of the two clusters, we can see that the individual CPU utilization of each cluster is well controlled at the reference, and the average CPU utilization of the base station is always quickly controlled to around 70 percent, which meets the real-time requirement.

8 Trace-Driven Simulations

In addition to the testbed experiments, we also conduct trace-driven simulations to extensively evaluate the performance of FAUC under a wide range of settings including network size and signal-to-noise ratio.

8.1 Simulation Methodology and Settings

The data traces used in the simulations were collected in the DARPA SensIT military vehicle localization experiment [24], where 75 WINS NG 2.0 nodes are deployed to localize Amphibious Assault Vehicles (AAVs) driving through a three-way intersection. We refer to [24] for detailed setup of the experiment. The data set used in our simulations includes the ground-truth data and the acoustic time series recorded by 17 nodes at a frequency of 4960 Hz. The ground-truth data include the positions of sensors and the trajectories of AAV runs recorded by GPS devices. However, as the vehicle localization experiment [24] only comprises a limited number of AAV runs, the data set cannot be readily used to evaluate the detection performance of FAUC. We note that, in order to evaluate the average error rate, the system needs to conduct enough detections in both cases of target presence and absence. To address this issue, we reuse the data traces as follows. In the simulations, we assume that the target appears at a fixed location within the deployment region. For each detection, an AAV run is randomly selected. In the presence of target, a sensor's measurement is set to be the sum of random noise and the real measurement when the AAV in the selected run is closest to the fixed location. In the absence of target, the sensor's measurement is only set to the random noise. Such a simulation methodology accounts for many realistic affecting factors such as terrain and the dynamical characteristics of the vehicle. Moreover, we can evaluate the performance of FAUC under different SNRs by changing the parameters of the noise generator. The target appearance probability P_a is set to be 50 percent. As there is no extra high-quality sensor such as camera in the SensIT experiment [24], we use a pseudocamera in the simulations, which generates random detection results based on the ground truth. The pseudocamera's false alarm rate and missing probability, i.e., P_{FH} and P_{MH} are set to be 2 percent. If the pseudocamera is activated in a detection, the CPU utilization in the detection follows a normal distribution with mean of 40 percent and standard deviation of 10 percent. The utilization bound u_s is set to be 35 percent.

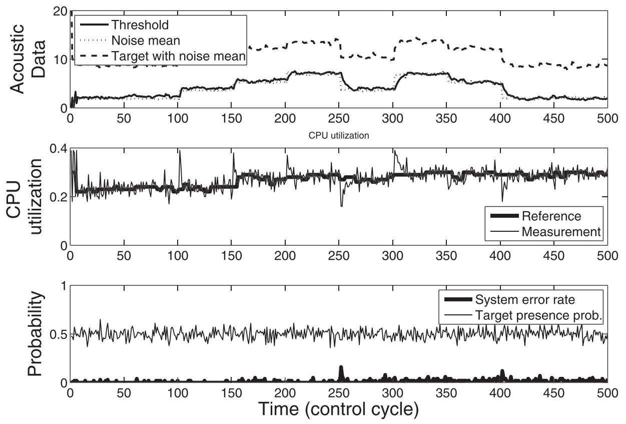


Fig. 16. The temporal evolution of the vehicle detection in the presence of changing noise (17 sensors).

8.2 Simulation Results

We first evaluate the temporal evolution of FAUC in the case of changing noise. The change of noise mean over time is shown in Fig. 16. The standard deviation of noise is initially set to be 0.1 and increased by 0.1 at the 200th and 350th control cycle. As we select a relatively high-utilization upper bound, the low-end sensors are allowed to raise more false alarms and the system can minimize the error rate. As a result, from Fig. 16, we can see that the calibrated threshold is around the noise mean. From the figure, we can also see spikes in utilization and error rate, which are caused by the changes of noise profile. However, the system can adapt to these changes within 20 control cycles.

In the second set of simulations, we evaluate the impacts of the number of sensors and SNR on the performance of FAUC. Fig. 17 shows the system error rate versus the number of sensors used in the simulations. Other settings are the same as in Fig. 16. We can see from the figure that the system detection performance increases with the number of sensors. Fig. 18 shows the system error rate versus SNR. Note that SNR is defined as $10\log_{10} S/\sigma$. We can see that the system detection performance increases with SNR. For both Figs. 17 and 18, the CPU utilization can be controlled to meet the upper bound u_s. Therefore, FAUC can work in wide range of settings of network size and SNR.

9 Conclusion

In this paper, we propose a holistic approach called *Fidelity-Aware Utilization Controller* to address both fidelity and timeliness requirements of Wireless Cyber-physical Surveillance systems. FAUC integrates data fusion with feedback control to enforce CPU utilization upper bound although system workloads vary significantly at runtime because of stochastic detection results. FAUC also optimizes system fidelity and adjusts the control objective of CPU utilization adaptively in dynamic environments. We have implemented FAUC on a small-scale WCS testbed consisting of TelosB/Iris motes and cameras. Our experiments on light and acoustic target detection show that FAUC can achieve robust fidelity and enforce desired utilization bounds in the presence of significant variations of target/noise characteristics and unreliable wireless links. Moreover, extensive simulations based on real acoustic data traces collected in a vehicle surveillance experiment show that FAUC can work in a wide range of settings including network size and SNR.

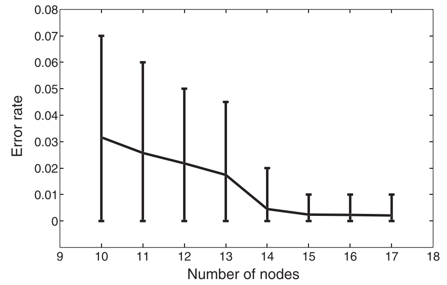


Fig. 17. System error rate versus the number of sensor nodes. The error bars are plotted with 5 and 95 percent quantiles.

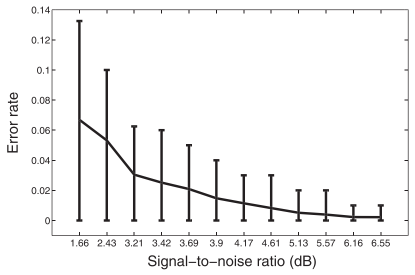


Fig. 18. System error rate versus SNR (17 sensors). The error bars are plotted with 5 and 95 percent quantiles.

Acknowledgments

This work is supported, in part, by the National Science Foundation under grants ECCS 1029683, CNS-0954039 (CAREER), and RC100824.

References

[1] Strix Systems Inc., http://www.strixsystems.com/, 2010.
[2] T. He, S. Krishnamurthy, J.A. Stankovic, T. Abdelzaher, L. Luo, R. Stoleru, T. Yan, L. Gu, J. Hui, and B. Krogh, "Energy-Efficient Surveillance System using Wireless Sensor Networks," *Proc. Second Int'l Conf. Mobile Systems, Applications, and Services (MobiSys)*, 2004.
[3] Memsic Inc., http://www.memsic.com, 2011.
[4] J.A. Stankovic, T. He, T. Abdelzaher, M. Marley, G. Tao, S. Son, and C. Lu, "Feedback Control Scheduling in Distributed Real-time Systems," *Proc. IEEE 22nd Real-Time Systems Symp. (RTSS)*, 2001.
[5] C. Lu, X. Wang, and X. Koutsoukos, "Feedback Utilization Control in Distributed Real-Time Systems with End-to-End Tasks," *IEEE Trans. Parallel Distributed Systems*, vol. 16, no. 6, pp. 550-561, June 2005.
[6] J. Yao, X. Liu, M. Yuan, and Z. Gu, "Online Adaptive Utilization Control for Real-Time Embedded Multiprocessor Systems," *Proc. Int'l Conf. Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, 2008.
[7] K. Whitehouse and D. Culler, "Calibration as Parameter Estimation in Sensor Networks," *Proc. First ACM Int'l Workshop Wireless Sensor Networks and Applications (WSNA)*, 2002.
[8] P.K. Varshney, *Distributed Detection and Data Fusion*. Springer, 1996.
[9] G. Xing, R. Tan, B. Liu, J. Wang, X. Jia, and C.-W. Yi, "Data Fusion Improves the Coverage of Wireless Sensor Networks," *Proc. MobiCom*, 2009.
[10] R. Tan, G. Xing, B. Liu, and J. Wang, "Impact of Data Fusion on Real-Time Detection in Sensor Networks," *Proc. IEEE 30th Real-Time Systems Symp. (RTSS)*, 2009.
[11] J. Feng, S. Megerian, and M. Potkonjak, "Model-Based Calibration for Sensor Networks," *Proc. IEEE Sensors*, 2003.

[12] R. Tan, G. Xing, X. Liu, J. Yao, and Z. Yuan, "Adaptive Calibration for Fusion-Based Wireless Sensor Networks," *Proc. IEEE INFOCOM,* 2010.

[13] X. Sheng and Y.-H. Hu, "Maximum Likelihood Multiple-source Localization Using Acoustic Energy Measurements with Wireless Sensor Networks," *IEEE Trans. Signal Processing,* vol. 53, no. 1, pp. 44-53, Jan. 2005.

[14] M. Hata, "Empirical Formula for Propagation Loss in Land Mobile Radio Services," *IEEE Trans. Vehicular Technology,* vol. 29, no. 3, pp. 317-325, Aug. 1980.

[15] D. Li and Y.-H. Hu, "Energy Based Collaborative Source Localization Using Acoustic Micro-Sensor Array," *EUROSIP J. Applied Signal Processing,* vol. 2003, pp. 371-375, 2003.

[16] C. Wren, U. Erdem, and A. Azarbayejani, "Functional Calibration for Pan-Tilt-Zoom Cameras in Hybrid Sensor Networks," *Multimedia Systems,* vol. 12, no. 3, pp. 255-268, 2006.

[17] P. Dutta, A. Arora, and S. Bibyk, "Towards Radar-enabled Sensor Networks," *Proc. ACM/IEEE Fifth Int'l Conf. Information Processing in Sensor Networks (IPSN),* 2006.

[18] T. He, P. Vicaire, T. Yan, L. Luo, L. Gu, G. Zhou, R. Stoleru, Q. Cao, J.A. Stankovic, and T. Abdelzaher, "Achieving Real-Time Target Tracking Using Wireless Sensor Networks," *Proc. IEEE 12th Real-Time and Embedded Technology and Applications Symp. (RTAS),* 2006.

[19] C.L. Liu and J. Layland, "Scheduling Algorithms for Multi-programming in a Hard Real-Time Environments," *J. ACM,* vol. 20, no. 1, pp. 46-61, 1973.

[20] C. Lu, X. Wang, and C. Gill, "Feedback Control Real-Time Scheduling in ORB Middleware," *Proc. IEEE Ninth Real-Time and Embedded Technology and Applications Symp. (RTAS),* 2003.

[21] J. MacQueen, "Some Methods for Classification and Analysis of Multivariate Observations," *Proc. Fifth Berkeley Symp. Math. Statistics and Probability,* 1967.

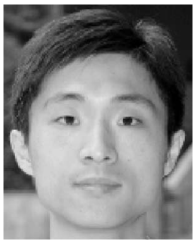
[22] K. Ogata, *Discrete-Time Control Systems.* Prentice-Hall, 1995.

[23] ImageMagick, http://www.imagemagick.org, 2010.

[24] M. Duarte and Y.-H. Hu, "Vehicle Classification in Distributed Sensor Networks," *J. Parallel and Distributed Computing,* vol. 64, no. 7, pp. 826-838, 2004.

Jinzhu Chen received the BS degree in communication engineering from University of Electronic Science and Technology of China, in 2005. From 2005 to 2009, he was an embedded software engineer at Delphi China Technical Center. He is currently working toward the PhD degree at Michigan State University. His research interests include wireless sensor networks and cyber-physical systems.

Rui Tan received the BS and MS degrees in automation from Shanghai Jiao Tong University, China, in 2004 and 2007, respectively, and the PhD degree in computer science from City University of Hong Kong in 2010. He is currently a postdoctoral research associate with Michigan State University. His research interests include collaborative sensing and computing in wireless sensor networks and cyber-physical systems.



Guoliang Xing received the BS degree in electrical engineering and the MS degree in computer science from Xian Jiao Tong University, China, in 1998 and 2001, respectively, and the DSc degrees in computer science from Washington University in St. Louis in 2006. He is an assistant professor in the Department of Computer Science and Engineering at Michigan State University. From 2006 to 2008, he was an assistant professor of computer science at City University of Hong Kong. He received the NSF CAREER Award and the Best Paper Award at the 18th IEEE International Conference on Network Protocols (ICNP) in 2010. His research interests include wireless sensor networks, mobile systems, and cyber-physical systems. He is a member of the IEEE.

Xiaorui Wang received the PhD degree from Washington University in St. Louis. He is an associate professor in the Department of Electrical and Computer Engineering at The Ohio State University. From 2006 to 2011, he was an assistant professor at the University of Tennessee, Knoxville. He is the recipient of the US Office of Naval Research (ONR) Young Investigator (YIP) Award in 2011, the US NSF CAREER Award in 2009, the Power-Aware Computing Award from Microsoft Research in 2008, and the IBM Real-Time Innovation Award in 2007. He also received the Best Paper Award from the 29th IEEE Real-Time Systems Symposium (RTSS) in 2008. His research interests include cyber-physical systems, power-aware computer systems and architecture, and real-time embedded systems. He is a member of the IEEE and the IEEE Computer Society.

Xing Fu received the BS and MS degrees from Beijing University of Posts and Telecommunications, China. He is currently working toward the MS degree in computer engineering in the Department of Electrical Engineering and Computer Science at the University of Tennessee, Knoxville. His research interests include power-aware data centers, embedded real-time systems, and cyber-physical systems. He is a student member of the IEEE.