Incorporating Emergency Alarms in Reliable Wireless Process Control

Bo Li[1]*, Lanshun Nie[2]*, Chengjie Wu[1]*, Humberto Gonzalez[3], Chenyang Lu[1]
[1]Department of Computer Science & Engineering, Washington University in St. Louis
[2]School of Computer Science and Technology, Harbin Institute of Technology
[3]Department of Electrical & Systems Engineering, Washington University in St. Louis

ABSTRACT

Recent years have witnessed adoption of wireless sensor-actuator networks (WSANs) in process control. Many real-world process control systems must handle various emergency alarms under stringent timing constraints in addition to regular control loops. However, despite considerable theoretical results on wireless control, the problem of incorporating emergency alarms in wireless control has received little attention. This paper presents, to the best of our knowledge, the first systematic approach to incorporate emergency alarms into wireless process control. The challenge in emergency communication lies in the fact that emergencies occur occasionally, but must be delivered within their deadlines when they occur. The contributions of this work are threefold: (1) we propose efficient real-time emergency communication protocols based on slot stealing and event-based communication; (2) we build an open-source WirelessHART protocol stack in the Wireless Cyber-Physical Simulator (WCPS) for holistic simulations of wireless control systems; (3) we conduct systematic studies on a coupled water tank system controlled over a 6-hop 21-node WSAN. Our results demonstrate our real-time emergency communication approach enables timely emergency handling, while allowing regular feedback control loops to effectively share resources in WSANs during normal operations.

Categories and Subject Descriptors

C.2.4 [**Distributed Systems**]: Distributed applications; C.2.1 [**Network Architecture and Design**]: Wireless communication

Keywords

Cyber-Physical System, Wireless Sensor-Actuator Network, process control, emergency alarms

1. INTRODUCTION

*The first three authors contributed equally to this work.

ICCPS'15 April 14–16, 2015, Seattle, WA, USA.
Copyright 2015 ACM 978-1-4503-3455-6/15/04 $15.00
http://dx.doi.org/10.1145/2735960.2735983.

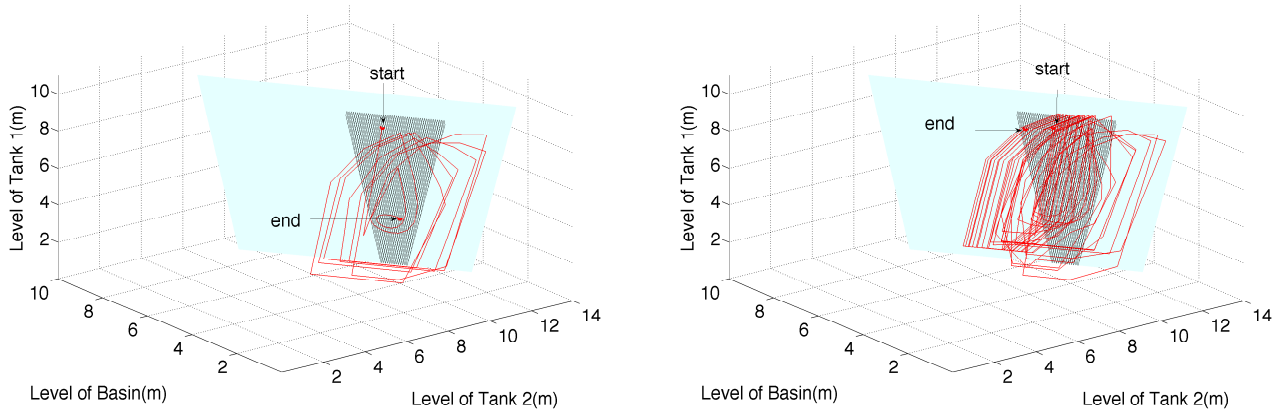
Wireless sensor-actuator network (WSAN) technology is gaining adoptions in process industries due to their advantage in lowering deployment effort in challenging environments. Industrial standard organizations such as ISA, HART, WINA and ZigBee, have been actively pushing the application of wireless technologies in industrial automation [15]. While early success of industrial WSANs focused on monitoring applications, there is significant value in exploring WSANs for process control applications to take full advantage of wireless technology in industrial plants.

A wireless process control system employs feedback control loops to control the dynamic response of industrial processes through communications in a shared WSAN. Since communication delays and packet drops may lead to severe degradation of control or even instability of the system, it is critical to support real-time and reliable communication.

Fig. 1 shows system state trajectories of wireless control versus ideal control for a water tank system. Here ideal control means the case where communications occur with no delay and no loss. Fig. 1(a) shows the ideal control system goes back to the shaded feasible region, reaches the set point and succeeds in control in a couple of rounds. In contrast, wireless control in Fig. 1(b) clearly takes more rounds and eventually fails to stabilize the system within the time limit, due to control packet drops and the communication delay. Hence, wireless control faces many challenges due to link failures and time varying delays (e.g., delay caused by retransmissions). In the face of emergencies, the control problem become even harder.

This work systematically investigates how to incorporate emergency alarms in wireless control systems, a problem that is critical in many real-world process plants, but yet received little attention in the literature. Simple controllers commonly used in industrial process control applications, such as PID or ON/OFF, can sometimes produce undesired responses, since they do not explicitly handle safety constraints. For this reason, it is also common to add safety measures, usually in the form of digital binary signals, to handle special situations that lead to physical damage of the plant, or even danger to the human operators. These signals take the form of tripwires around dangerous zones, emergency triggers for human operators, or contact switches in water tanks, among many others. In a wireless control scenario, as the one described in this paper, these emergency signals must be transmitted using the same infrastructure as the regular control signals.

Despite significant body of theoretical results on real-time



(a) Ideal Control

(b) Wireless Control

Figure 1: Control State Trajectory of A Coupled Water Tank System: Ideal vs. Wireless

communication protocols and scheduling for WSANs, earlier research has largely focused on regular feedback control loops that employ communications in a periodic or event-driven fashion. Emergency alarms presents a challenging communication and control design problem. While emergencies occur sporadically, it is critical to communicate and handle emergency alarms in a timely fashion when they happen. Moreover, the lack of realistic simulation tools in compliant with state-of-art WSAN standards (e.g., WirelessHART [1]) has largely prevented in-depth wireless process control research. In this paper we present the following contributions to address these challenges:

- We implement an WirelessHART protocol stack in the TOSSIM wireless simulator, on top of its realistic link model for IEEE 802.15.4 radios.
- We build the Wireless Cyber-Physical Simulator 2.0 that integrates Simulink and TOSSIM for holistic wireless control study while supporting both periodic and event-based simulations.
- We propose periodic and event-based real-time emergency communication protocols for WSAN.
- We construct a systematic case study on a coupled water tank system controlled over a 6-hop WSAN.

The rest of the paper is organized as follows. Section 2 discusses related works. Section 3 presents the system model of a wireless control system. Section 4 introduces the WCPS 2.0 simulator, the WirelessHART protocol stack and the emergency communication protocols. Section 5 details the control design for the case study. Section 6 presents the case study and evaluation of the proposed approaches. Section 7 concludes the paper.

2. RELATED WORK

Promising results have been reported in the wireless control literature. Case studies on wireless structural monitoring and control systems were reported in [20,21,27,28]. Real-time transmission scheduling and co-designs for WSANs has been investigated in [11, 24, 25, 31]. Reliable routing algorithms for WSANs have been presented [15]. Unfortunately none of these works considered emergency alarms.

Networked control systems have received tremendous attentions [9]. Discrete-time Kalman filters have been proposed for state estimation based on intermittent observation [26]. Co-design of transmission scheduling and controllers was explored in [13]. Passivity-based control architecture was proposed for cyber-physical systems [17]. Fault-tolerant control under uncertainties and time delays was studied in [12]. These work did not consider emergency alarms either.

Progress on WSAN protocols have been reported. Self-triggered control approaches have been developed for wireless networks [7, 29]. A distributed control approach has been proposed for WSANs [22]. These works however focused only on regular feedback control loops. Our work complements them by investigating emergency alarms alongside regular feedback loops.

Because large-scale real-world wireless control systems are not always available, a number of simulation tools have been developed. Truetime [4] is a well established control system simulator that enables holistic studies of CPU scheduling, communication and control algorithms. NCSWT [14] is a useful simulator for wireless cyber-physical systems. None of these simulators implemented WirelessHART, which is widely used in the industry. Gisso in [6] is a recent simulator for wireless control systems based on Cooja, but the wireless link model in Cooja simulation remains to be improved. WCPS [19] connects Simulink and TOSSIM. WCPS 2.0 as a further development in this paper has incorporated substantial changes including a new WirelessHART protocol stack. Finally, WCPS 2.0 can effectively simulate aperiodic emergency events.

Despite the fact that fault detections have been heavily studied in wireless sensor network [10, 32] and process control [5, 30], efforts in this study are orthogonal to existing fault tolerant literatures because those efforts mostly detect and isolate faults caused by sensor or controller failures rather than wireless link failures. Challenges arising from wireless link failures remain a problem even after detection and isolation of sensor or control failures. As such, reliable network protocols in this study is a natural complement for existing fault tolerant literature.

3. SYSTEM MODEL

We consider a wireless control system consisting of a physical plant, a centralized controller and a WSAN. Sensors and actuators communicate through a multi-hop WSAN forming a multi-hop wireless mesh network. In the *sensing phase*, sensors send their measurements to the controller. Control commands issued by the controller will be sent to actuators in the *actuation phase* through the same WSAN.

There are two types of flows in our system: periodic regular flows and aperiodic emergency alarms. A regular flow generates packet periodically in both *sensing phase* and *actuation phase*. Emergency alarms are triggered sporadically. Packets of a regular flow or an emergency alarm must be delivered within its deadline. An emergency alarm is more critical than a regular flow.

Based on the state-of-art WirelessHART standard [1], the WSAN adopts a centralized architecture in our design. The Network Manager and Access Points are usually connected by reliable wired links while the rest of the WSAN communicate using the wireless mesh network. The transmission schedule is organized in terms of time slots (10 ms per slot). The network protocol stack comprises (1) a routing layer that supports both source routing and reliable graph routing. (2) a MAC layer running a multi-channel Time Division Multiple Access (TDMA) protocol and (3) the IEEE 802.15.4 physical layer for low-power radios.

4. WIRELESS DESIGN

In this section, we firstly introduce our Wireless Cyber-Physical Simulator [2] [19]; we then describe our WirelessHART stack implementation; we finally we present our real-time emergency communication protocols, and other major changes in WCPS 2.0.

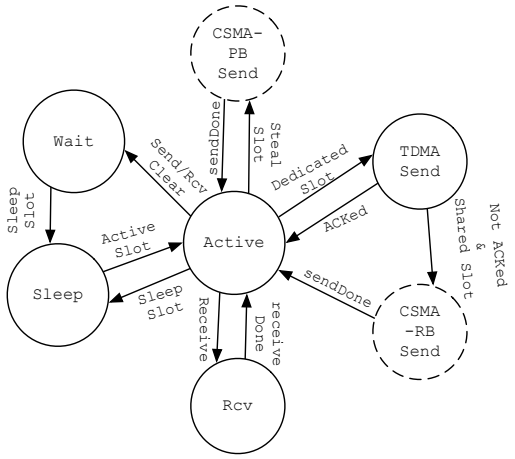


Figure 2: Finite State Machine for Wireless Sensors

4.1 WCPS 2.0

To support holistic cyber-physical co-design and evaluation of wireless control systems we have developed Wireless Cyber-Physical Simulator (WCPS), an integrated simulator for wireless control systems. Simulink has been widely used for control system designs; TOSSIM is designed to simulate wireless sensor networks based on a realistic wireless link model validated in diverse real-world environments [18]. WCPS 1.0 has employed a federated architecture and integrated Simulink and TOSSIM.

In this work, we have substantially extended WCPS to version 2.0 by implementing a WirelessHART protocol stack comprised of multi-channel communication, reliable Graph Routing and Dedicated/Shared time slotting supported by a robust Finite Sate Machine, and a centralized TDMA scheduler. Moreover, to accurately simulate aperiodic events, we have reorganized the simulator architecture to support event-driven co-simulation between TOSSIM and Simulink.

Under the new co-simulation architecture, TOSSIM is configured as a TCP/IP server that simulates the WSAN. Control models in Simulink connects to the TOSSIM server as a socket client. Since TOSSIM is a discrete event simulator, 10^7 event ticks(time steps) corresponds to 1ms. In our co-simulation, each client call from Simulink will advance TOSSIM by 10ms (i.e., 10^8 ticks). Configuring TOSSIM as a background server process allows effective data exchange between TOSSIM and Simulink while preserving all system states across client calls. To our knowledge, WCPS 2.0 is the first simulator that can simulate high-fidelity interaction between TOSSIM and Simulink.

More details about WCPS (including user manual, documentation and the source code) are available at `http://wcps.cse.wustl.edu`.

4.2 WirelessHART Stack

As the WirelessHART standard is gaining widespread adoption in process industries, it is important to study wireless control systems based on WirelessHART networks. As an integral part of WCPS 2.0, we have implemented a WirelessHART protocol stack in the TOSSIM simulator. Our WirelessHART stack realizes WirelessHART protocols at the routing and MAC layers, extends the TOSSIM link model to support multiple channels, and also implements a centralized network manager with a routing algorithm and transmission scheduler. To our knowledge, WCPS 2.0 is the first simulator that supports all these WirelessHART features.

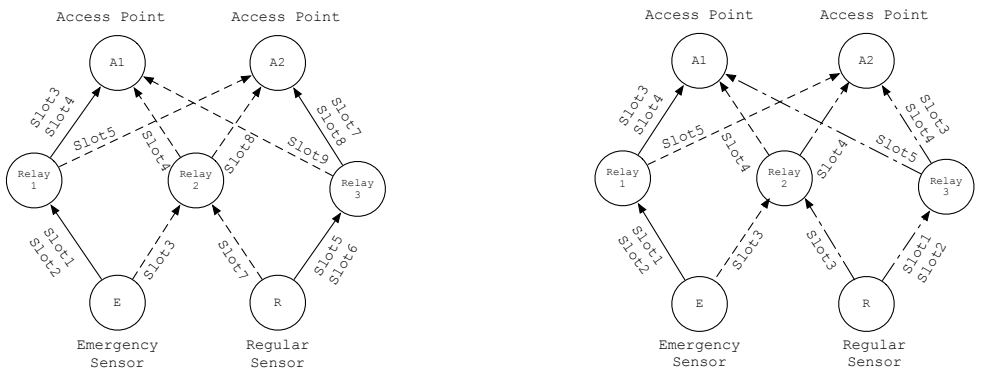
4.2.1 Multi-channel Communication

The original TOSSIM wireless model only supports a single channel. A key feature of WirelessHART is exploiting spectrum diversity by utilizing multiple channels supported by IEEE 802.15.4 radios. To support WirelessHART networks, we extend the TOSSIM simulator to support communication over multiple channels. The extended TOSSIM in WCPS 2.0 now can accept wireless signal and noise traces of multiple channels simultaneously and use them as inputs for simulations of wireless communication over multiple channels within a same time slot.

4.2.2 Graph Routing

WirelessHART supports two types of routing, Source Routing and Graph Routing, the latter of which is desirable for reliable communications (e.g., emergency alarms). Source Routing provides a single route for each sensor/actuator; Graph Routing improves reliability through redundant routes, where each node in a graph route has two alternative receivers. In Graph Routing, two *dedicated slots* are first allocated for transmissions to the primary receiver, followed by a *shared slot* for the retransmission to the alternative receiver.

While WCPS 1.0 only supported Source Routing, we have implemented Graph Routing in TOSSIM for WCPS 2.0. This new routing approach enables us to explore reliable communication for wireless control systems. Further, we have implemented a robust Finite State Machine(FSM, see



(a) Periodic Scheduling (PS) (b) Slot Stealing (SS)

Figure 3: Wireless Communication Protocols

Fig. 2) that runs in the MAC layer of wireless sensors, which supports execution of Graph Routing, dynamic channel hopping, and TDMA schedules.

4.2.3 Dedicated/Shared Time Slotting

We implement both dedicated and shared slots at the MAC layer. In a *dedicated* slot, only one transmission is allowed on a same channel; in contrast, multiple transmissions can be scheduled in a same channel to contend in a *shared* slot. As shown in Fig. 2, in a dedicated slot, the owner does TDMA Send without channel assessment. On the other hand, CSMA with Random Backoff (CSMA-RB) is used in *shared* time slots, when different sensors compete for the transmission opportunity. TDMA Send and CSMA-RB can provide basic supports for WirelessHART communications.

We further devise CSMA Permanent Backoff (CSMA-PB) to support Slot Stealing. Sensors that send packets with CSMA-PB will *permanently* cease any transmission attempt within a slot when energy of others has been detected. That is, during Slot Stealing, *owner* will send a packet at the very beginning of the slot while the *stealer* will do channel assessment with an offset, followed by a backoff if channel is not clean, or a transmission otherwise.

4.2.4 Centralized Scheduler

We have also implemented a centralized network manager including a graph routing algorithm and a transmission scheduler. The transmission scheduler generates a superframe consisting of a sequence of time slots, each assigned a set of transmissions to occur on different channels. At run time the schedule is used in a cyclic fashion, repeating the superframe after reaching the end of the schedule.

4.3 Real-time Emergency Communication

Given a WirelessHART network, we consider the real-time communication problem of $k+l$ flows $F = \{E_1, .., E_k, R_1, ..., R_l\}$. Each regular flow $R_i \in R$ is periodically generated with a period P_i and a deadline D_i, where $D_i \leq P_i$. As specified in Graph Routing [15], from the source to the destination, there exists at least *two* outgoing links(one primary, one backup) for every non-destination node. An emergency flow $E_j \in E$ is triggered aperiodically with a deadline D_j. The communication latency L_n of a packet for a flow generated at slot n and delivered at slot m is defined as $m - n + 1$.

We observe the real-time communication problem for wireless control involve mixed criticalities, where the emergency flows have higher criticality than regular flows. The objectives of real-time communication are two fold: (1) In the regular mode, i.e., when there is no emergency, all regular flows should meet their deadlines; (2) In the emergency mode when emergency occurs, emergency flows should meet their deadlines, while no guarantee is provided to regular flows.

The challenge in supporting emergency communication lies in the fact that emergencies occur only occasionally and the system operates in the regular mode most of the time. However, when emergency does occur, it is critical to meet the deadlines of emergency flows.

A simple approach to schedule an emergency flow is to reserve time slots for a virtual periodic flow (also called a *periodic server*) that is scheduled alongside the regular flows. Emergency alarms are transmitted within the time slots designated to the periodic server. A drawback of this periodic scheduling (PS) approach is that it wastes network bandwidth when there is no emergency.

To avoid wasting resources during the regular mode, we introduce a slot stealing (SS) mechanism that allows regular flows to *steal* slots from emergency schedule when emergency does not exist, and thus would enhance slot utilization during regular operations. Furthermore, we propose event-based emergency communication to further improve network efficiency during the emergency mode.

4.3.1 Periodic Scheduling (PS)

PS creates a virtual periodic flow for each emergency alarm and schedule them alongside the regular flows. Emergency alarms are transmitted in the slots allocated for the corresponding virtual periodic flow.

We adopt a fixed priority scheduling policy and a two-level priority assignment approach. Virtual periodic flows always have higher priorities than regular flows. Among the virtual periodic flows, we assign their priorities based on the rate monotonic policy. Similarly, regular flows are also prioritized based on the rate monotonic policy.

For example, Fig. 3(a) illustrates a transmission schedule of PS. For simplicity purposes, this example uses a single channel, but we consider multi-channel communication in our case studies. Links in Fig. 3(a) are categorized as primary paths (solid lines) and backup paths (dashed lines). Communication on primary paths happens in *dedicated* time slots while communication on backup paths are scheduled in *shared* time slots, when different senders may contend for transmission opportunities. Emergency sensor E is sched-

uled to transmit to Relay 1 in Slot 1 and Slot 2 . If either of the transmissions in Slot 1 or 2 succeeds, following transmissions from Relay 1 to A1 will be scheduled in Slot 3 and Slot 4. However, if both transmissions in Slot 1 and Slot 2 fail, a backup link will be used by E to transmit to Relay 2 and then from Relay 2 to A1, in shared Slot 3 and Slot 4, respectively. Data from the regular sensor R will take similar scheduling and routing strategy. PS takes 9 slots in total to schedule both flows. Algorithm 1 shows a detailed algorithm of PS.

```
   input  : E, R, routes, connectivity
   output: S[1 ··· T][0 ··· m − 1]
 1 F ← {E, R}; ch ← 0; m ← total channel; T ←hyper
   period;
 2 while (F ≠ ∅) do
 3 |   flo ← Highest priority flow in F; rout ← {route
   |   of flo} ⊂ routes;
 4 |   while (rout ≠ ∅) do
 5 |   |   send ← first transmission on rout.
 6 |   |   if (s ≤ T) then
 7 |   |   |   if ({conflicts in connectivity}= ∅) then
   |   |   |   S[s][ch] ← send; ch ← ch + 1;
 8 |   |   else
 9 |   |   |   return unschedulable.
10 |   |   end
11 |   |   rout ← rout − {send}; s ← s + 1;
12 |   end
13 |   F ← F − {flo};
14 end
15 return S[1 ··· T][0 ··· m − 1];
```

Algorithm 1: Periodic Scheduling(PS)

4.3.2 Periodic Scheduling with Slot Stealing (SS)

In PS, time slots allocated to emergency flows are left unused when there is no emergency, which is a waste of precious network resource. To overcome this limit, SS allows emergency alarms and regular flows to be scheduled in the *same* dedicated slots. When emergency does not exist, emergency slots will be used(stealed) by regular flows instead. Whenever an emergency exists, the emergency transmission would take the slot while the regular transmission would back off.

Slot Stealing is technically inspired by hybrid MAC protocols such as Z-MAC [23]. An emergency packet is transmitted immediately at the beginning of a slot shared with the regular packet. In contrast, a regular packet first performs a Clear Channel Assessment(CCA) after waiting for a constant backoff time. If there is any other transmission going on(likely from an emergency sender), the regular sender would cease its transmission. Otherwise, it goes ahead and transmit the packet.

Fig. 3(b) shows an example of SS. Following the same retransmission and Graph routes as PS, we see SS takes 5 time slots (4 slots fewer) to accommodate both flows. Algorithm 2 depicts the detailed algorithm of SS.

4.3.3 Event-based Slot Stealing (SS-Event)

There are two alternative approaches to send emergency alarms during an emergency. For systems that need to periodically monitor and control the emergency state, an emergency control flow is activated whenever emergencies exist. The emergency flow then periodically generates sensor data and control command until the emergency is over.

For systems that do not need to periodically monitor and control the emergency state, the system can adopt an event-based approach to communicate the emergency alarms, i.e., an emergency sensor only sends an alarm-start and an alarm-end packets in the beginning and end of the emergency. While this event-based communication results in the same transmission schedule as in Algorithm 2, event-based SS communication can significantly reduce the number of regular transmissions that are affected by emergency transmissions, potentially leading to better control performance. Hence forth, we denote the combination of event-based communication and SS as SS-Event.

We note SS and SS-Event have clear tradeoffs between data loads and communication reliability. Periodic flows in SS on one hand would reduce chances of missing emergency alarms while on the other it would override regular flows with excessive periodic traffics(in stealed slot). In contrast, SS-Event has less impact on regular flows but runs at the danger of completely missing critical alarm packets.

```
   input  : E, R, routes, connectivity
   output: S[1 ··· T][0 ··· m − 1]
 1 ch ← 0; m ← total channel; T ←hyper period;
 2 Schedule E with the PS algorithm.
 3 while (R ≠ ∅) do
 4 |   flo ← Highest priority flow in R; rout ← {route
   |   of flo} ⊂ routes;
 5 |   while (route ≠ ∅) do
 6 |   |   send ← first transmission on rout.
 7 |   |   if (s ≤ T) then
 8 |   |   |   if ({conflicts in connectivity}= ∅) then
 9 |   |   |   |   if ({free channel} ≠ ∅) then
   |   |   |   |   S[s][ch] ← send; ch ← ch + 1;
10 |   |   |   |   else ch ← sharable ch of emergencies;
   |   |   |   |   S[s][ch] ← send;// Steal
11 |   |   |   else
12 |   |   |   |   if ({shareable channel} ≠ ∅) then
   |   |   |   |   ch ← sharable ch of emergencies;
   |   |   |   |   S[s][ch] ← send;// Steal
13 |   |   |   |   else return unschedulable;
14 |   |   |   end
15 |   |   else
16 |   |   |   return unschedulable.
17 |   |   end
18 |   |   rout ← rout − {send}; s ← s + 1;
19 |   end
20 |   R ← R−{flo};
21 end
22 return S[1 ··· T][0 ··· m − 1];
```

Algorithm 2: Scheduling with Slot Stealing

5. CONTROL DESIGN

We apply our emergency handling protocol in a coupled water tank system as a case study. In this section we describe the dynamical model of the water tank system and our controller design.

5.1 Coupled Water Tank

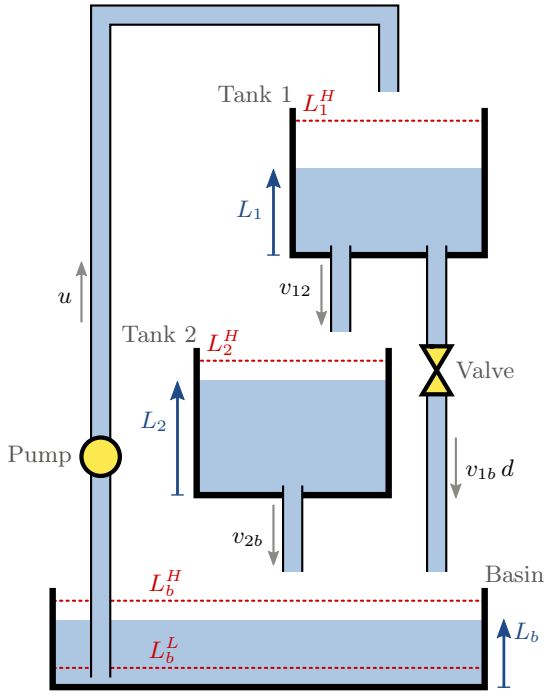


Figure 4: Diagram of the coupled water tank system. The water levels of Tank 1, Tank 2, and Basin are denoted L_1, L_2, and L_b, respectively. The emergency water levels are denoted L_1^H, L_2^H, L_b^L, and L_b^H. The natural pipe flows are denoted u, v_{12}, v_{1b}, v_{2b}. The state of the valve is denoted $d \in \{0, 1\}$, where $d = 1$ if the valve is *ON*.

A diagram of the coupled water tank is shown in Figure 4. This system shares similar dynamics with many other process control systems, e.g., irrigation networks [7]. Our choice to use this system as a case study is based on its simple yet representative dynamics, its hybrid dynamical nature (as the evolution of the system changes when the water tanks are either full or empty), and more importantly, its similarity to systems commonly used in industrial applications.

The coupled water tank system is comprised of one pump, one ON/OFF valve, two water tanks, and one basin. The pump is responsible for pushing water from the basin to Tank 1. The flow through the pump is a controlled variable, denoted u. Tank 1 is placed higher than Tank 2, and water flows due to gravity via a pipe at the bottom of Tank 1 placed above Tank 2.

The flow through this pipe is denoted v_{12}, and satisfies the following equation:

$$v_{12} = \frac{1}{\rho\, R_{12}} \sqrt{\rho\, g\, L_1}, \tag{1}$$

where ρ is the density of water, g is the gravity constant, R_{12} is the resistance parameter of the pipe, and L_1 is the level in Tank 1.

Similarly, Tank 2 is placed higher than the basin, and water flows via a pipe at the bottom of Tank 2 placed above the basin. The flow through this pipe is denoted v_{2b}, and satisfies the following equation:

$$v_{2b} = \frac{1}{\rho\, R_{2b}} \sqrt{\rho\, g\, L_2}, \tag{2}$$

where, besides the parameters defined in Equation (1), R_{2b} is the resistance parameter of the pipe, and L_2 is the level in Tank 2.

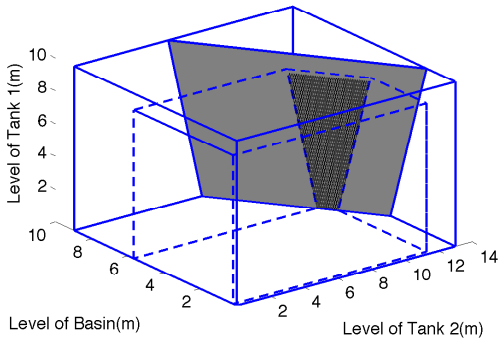


Figure 5: Diagram of the state space of the coupled water tank system. The gray plane corresponds to the feasible states since no water enters or exits the system. The dark region of the plane corresponds to the region where no emergencies occur. The dotted lines show emergency limits, and the solid lines show the physical limits of the tanks.

A pipe at the bottom of Tank 1, above the basin, is interrupted by the ON/OFF valve, hence water flows only when the valve is ON. The flow through this pipe, denoted v_{1b} is zero when the valve is OFF, and satisfies the following equation when the valve is ON:

$$v_{1b} = \frac{1}{\rho\, R_{1b}} \sqrt{\rho\, g\, L_1}, \tag{3}$$

where R_{1b} is the resistance parameter of the pipe.

Using conservation of mass and equations (1) to (3) we get the following dynamic equations for the coupled water tank system:

$$\frac{\mathrm{d}L_1}{\mathrm{d}t} = \begin{cases} \frac{1}{\rho\, A_1}\left(-v_{12} - v_{1b}\, d + u\right) & \text{if } L_1 \in [0, L_1^{\max}], \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

$$\frac{\mathrm{d}L_2}{\mathrm{d}t} = \begin{cases} \frac{1}{\rho\, A_2}\left(v_{12} - v_{2b}\right) & \text{if } L_2 \in [0, L_2^{\max}], \\ 0 & \text{otherwise,} \end{cases} \tag{5}$$

$$\frac{\mathrm{d}L_b}{\mathrm{d}t} = \begin{cases} \frac{1}{\rho\, A_b}\left(v_{1b}\, d + v_{2b} - u\right) & \text{if } L_b \in [0, L_b^{\max}], \\ 0 & \text{otherwise,} \end{cases} \tag{6}$$

where $L_1^{\max}$, $L_2^{\max}$, and $L_b^{\max}$ are the physical heights of the tanks, A_1, A_2, and A_b are the areas of the tanks, and $d \in \{0, 1\}$ is a controlled variable such that $d = 1$ when the valve is ON, and $d = 0$ when the valve is OFF.

5.2 System Emergencies

In our case study, the objective is to achieve set-point tracking of the water level in the Tank 2 by adjusting the flow u. We assume that Tank 2 has a level sensor (measuring L_2), and that the pump allows us to fully control the flow $u \in [0, u^M]$.

We also define four emergency situations, three of them corresponding to each water tank having too much water, which may produce spillage, and one corresponding to the basin having too little water, which may lead to the pump sucking air instead of water. Using the notation in Figure 4, the emergencies occur in the following situations: $L_1 > L_1^H$, $L_2 > L_2^H$, $L_b > L_b^H$, and $L_b < L_b^L$.

Note that the coupled water tank system is closed, i.e., it only recirculates water, and no water enter or leaves the system. This condition can be observed from equations (4) to (6), since $A_1 \frac{dL_1}{dt} + A_2 \frac{dL_2}{dt} + A_b \frac{dL_b}{dt} = 0$ (when the water levels are within the normal limits). In other words, $A_1\, L_1(t) + A_2\, L_2(t) + A_b\, L_b(t) = A_1\, L_1(0) + A_2\, L_2(0) + A_b\, L_b(0)$ for each $t \geq 0$. Using this extra constraint, even though the system has three states, we can plot its trajectories in a two-dimensional plane, as shown in Figure 5. The dark region in that figure corresponds to the subset of the two-dimensional plane where no emergencies occur. The rest of the two-dimensional plane corresponds to the state space where at least one emergency is active. The two-dimensional plane is bounded by the physical constraints of the system, i.e., the fact that all levels must remain above zero and below the maximum height.

5.3 Actuator and Controller Design

For water level control in Tank 2 we use a PID controller sensing L_2 and acting on u. Also, to efficiently correct emergencies, the valve, which is normally *OFF*, is sometimes switched to *ON*. Thus, the control strategy for this system is hybrid, since whenever an emergency is activated the controller behavior is changed. PID parameters are decided empirically in this study, we refer interested users to our code for more design details.

The PID controller, used when no emergencies are active, follows the following equations:

$$u(t) = K\left(e(t) + \frac{1}{T_i}\int_0^t e(s)\,ds + T_d\,\frac{\mathrm{d}e}{\mathrm{d}t}(t)\right), \qquad (7)$$

where $e(t) = L_2^{\mathrm{sp}} - L_2(t)$, L_2^{sp} is the desired set-point, and K, T_i, and T_d are the controller parameters. Also, whenever the right-hand side of equation (7) is above u^M then we set $u(t) = u^M$, and when it is below zero then we set $u(t) = 0$. The interested readers can go to [8] for more details regarding PID controllers.

We apply the following rules when emergencies are activated, in priority order:

1. If $L_1 > L_1^H$, $L_2 > L_2^H$, or $L_b < L_b^L$, we set $u = 0$ and $d = 1$ (i.e., shut off the pump and open the valve).
2. If $L_b > L_b^H$, and either $L_1 > L_1^H$ or $L_2 > L_2^H$, then we set $u = 0$ and $d = 1$.
3. If $L_b > L_b^H$, we set $u = u^M$ and $d = 0$ (i.e., pump as much water as possible from the basin and close the valve).

The rules above are heuristics designed under the assumption that maintaining the level in Tanks 1 and 2 is more important than maintaining the level in the basin. Hence, if either Tank 1 or Tank 2 have too much water, or if the basin has too little water, we transport water from the Tanks to the basin as quickly as possible (Rule 1). Since removing water from either of the Tanks conflicts with removing water from the basin, the Tanks take precedence (Rule 2). Finally, if the water level in the basin is too high then we pump water from the basin as quickly as possible (Rule 3).

To avoid high frequency switching between different emergency modes (or even Zeno executions [16]), the controller is forced to stay at least $\kappa > 0$ seconds in each mode after it is activated.

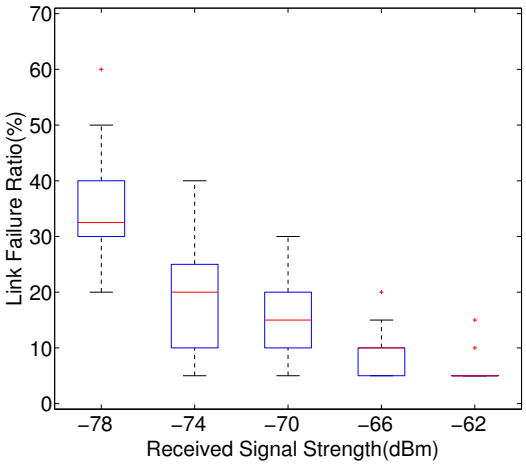


Figure 6: Link Failure Ratio

6. CASE STUDY

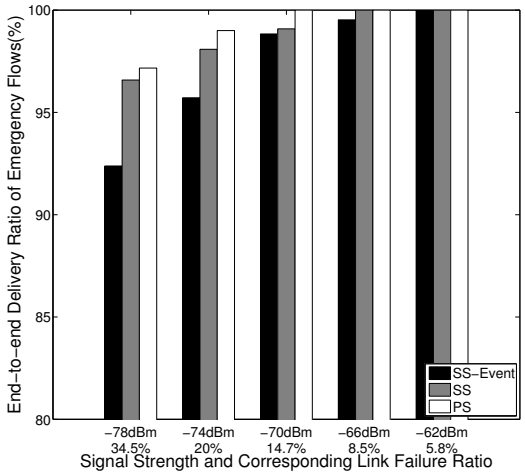
This section presents performance evaluations with a case study. The objective here is not simply to compare networking protocols. Rather, we would like to systematically evaluate our holistic framework, including the WCPS 2.0 simulator, the WirelessHART stack, as well as the real-time communication protocols.

To stress the WSAN, we have implemented two sets of Coupled Water Tank systems sharing the same WSAN. Each coupled water tank set is comprised of two water tanks and one basin. In total we have 4 tanks and 2 basins in the plant. We have attached 8 emergencies sensors and 2 regular sensors for monitoring purposes. In our system, plant data is first generated by water tanks and then fed into the WSAN in TOSSIM. Having been transmitted through the WSAN, sensor data with delay and loss information will be updated to the controller. Control commands from the controller would later be transmitted through the downlink WSAN and eventually applied for closed-loop control. We note since WirelessHART adopts deterministic TDMA schedules, in this evaluation we focus more on impact of network reliabilities instead of network delays; for sensor data that have not arrived before the deadline (e.g., dropped or missed the deadline), control decisions are made upon most recent available packet from the same sensor.

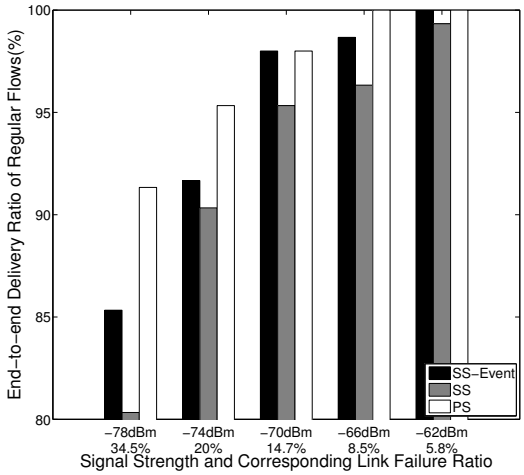
6.1 Network Performance

In our study, wireless traces from a 21-node subset of the Wustl Testbed [3] have been used to form a WSAN in TOSSIM, with a maximum path length of 6 hops . We directly use the connectivity and the multi-channel wireless noises from the Wustl Testbed; on the other hand, we use controlled Received Signal Strength with uniform gaps to simulate various wireless signal strength. Fig. 6 shows link quality statistics (3000 packets), where -74 dBm and -62 dBm features 20% and 5.8% averaged link failure under the Wustl Testbed noise, respectively.

Fig. 7 shows end-to-end delivery ratios of the three communication protocols implemented in WCPS 2.0. Since all experiments are done with the Wustl Testbed noise, we show both Received Signal Strength and corresponding Link Failure Ratio on the x-axis. In other words, multi-hop end-to-end delivery ratios in Fig. 7 can be reproduced in WCPS



(a) Delivery Ratio for Emergencies



(b) Delivery Ratio for Regular Flows

Figure 7: Average End-to-end Delivery Ratio

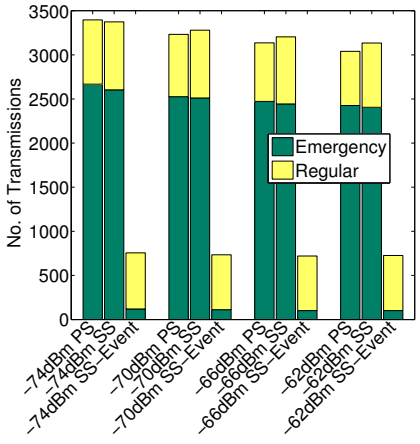


Figure 8: Number of Transmissions under Various Wireless Conditions

2.0 as long as Link Failure Ratio is the same, whereas signal strength and noise traces needn't. As in Fig. 7(a) and Fig. 7(b), both types of flows have better end-to-end delivery ratios as link failures improve(Received Signal Strength increases).

It is interesting to see PS, SS and SS-Event all achieve over 95% emergency delivery and 90% regular flow delivery ratios at 20% link failure ratio (-74dBm), implying that Graph Routing is working properly. Note for all 3 protocols, emergencies in Fig. 7(a) outperforms regular flows in Fig. 7(b), which is because emergency flows always have higher transmission priorities over regular ones. If we compare SS against SS-Event in Fig. 7(a) and Fig. 7(b), respectively, we can see SS has better emergency alarm delivery ratios (because of redundant periodic flows) and worse regular flow deliveries (because of conflicts from emergencies).

Fig. 8 further shows number of transmissions (50 second simulation) under various wireless conditions. Dark part of a bar means transmission counts for emergencies while light parts represent traffics for regular flows. As expected, PS and SS have more emergency transmissions due to the periodic nature. For regular traffics, it is interesting to see that SS has more regular traffics than the other two, which is because of retransmissions caused by backoffs during Slot Stealing. This also validates correctness of our WirelessHART stack and the Slot Stealing mechanism.

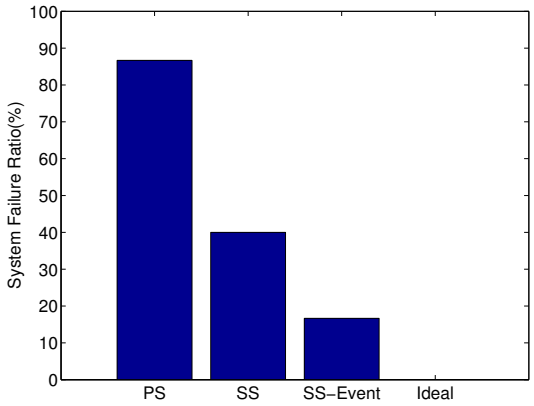
6.2 Control Performance

In our control evaluations, we choose to study cases with challenging initial conditions(e.g., small trapezoid area in Fig. 5). System failure is defined in twofold: first, the system can not stabilize inside feasible region given a time bound, e.g., 100 seconds; second, the system violates physical constraints, e.g., water spilling.

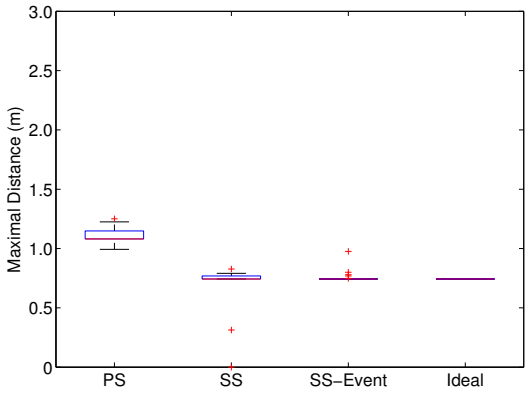
We further adopt three general evaluation metrics: *system failure ratio*, *time percentage outside the feasible region*, and *maximum distance to the feasible region*. *Time percentage outside the feasible region* is defined as the time percentage when the system state is out of the feasible region but within the control time limit. *Maximum distance to the feasible region* is defined as the maximum distance of system state to the feasible region(see Fig. 5). Simulations were executed under the following conditions:

- No emergency exists at the beginning;
- Total amount of water (i.e. $A_1\,L_1(0) + A_2\,L_2(0) + A_b\,L_b(0)$) is equivalent to maximum allowed capacity of Tank 2 and Basin ($A_2\,L_2^H + A_b\,L_b^H$);
- Emergency control runs at maximally supported frequency by the WSAN.
- Each simulation lasts 200 seconds, decided by the system time constant.

In our case study, we set control period consistently as 1Hz for regular loops while setting emergency control on the *maximally supported frequency*, bounded by the operation period of the WSAN. For example, TDMA superframe for SS-Event has 48 time slots and hence the maximally supported operation frequency for SS-Event is 2Hz(i.e., 500ms period). Similarly, PS who has a longer superframe can only operate at a slower control frequency, i.e., 1Hz in our case study.



(a) System Failure Ratio



(b) Maximal Distance to Feasible Region of Tank 2

Figure 9: Control Performances under 20% Link Failures (-74dBm Wireless Signal Strength)

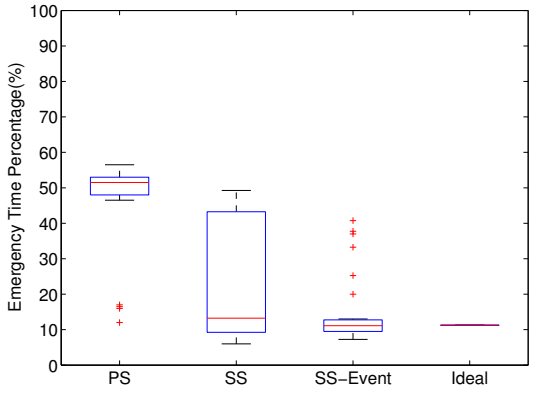


Figure 10: Emergency Time Percentage under 20% Link Failures (-74dBm Wireless Signal Strength)

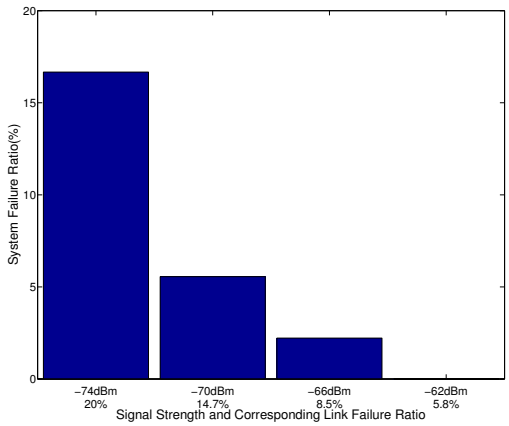


Figure 11: System Failure under Various Wireless Conditions

All statistics in Fig. 9 are done under 20% link failures (-74dBm wireless signal strength), with results averaged from 15 simulations (3000 control steps). System failure ratios in Fig. 9(a) shows SS-Event has achieved the closest performance compared to *Ideal(Wired) Control*, which is because SS-Event has higher emergency control frequency and hence shorter delays. A close look shows that SS is less successful than SS-Event is because SS has consistently dropped too many regular packets due to conflicts.

Maximum distance to the feasible region of Tank 2 in Fig. 9(b) and time percentage outside the feasible region in Fig. 10 shows the same trend that SS-Event is the better than the other two. Fig. 11 further depicts control performance of SS-Event under various wireless conditions. For cases featuring 5.8% link failures, SS-Event have 0% system failure, i.e., 100% control success. This is indeed encouraging as it demonstrates even for a 6-hop lossy wireless network, successful system control can still be achieved with careful wireless designs.

In sum, sitting on top of the state-of-art WirelessHART protocol stack in WCPS 2.0, we for the first time have been able to do scalable case studies for wireless emergency handling, and encouraging experiment results have been achieved. The periodic and event-based communication framework can be easily tailored according to application needs or optimization formulations, which is beyond the scope of this work and left for future studies.

7. CONCLUSION

Recent years have witnessed significant interests in adopting wireless sensor-actuator networks in process control. However, the problem of incorporating emergency alarms in wireless process control remains to be explored. This paper presents the first systematic approach to integrate emergency alarms into wireless process control systems. The challenge in emergency communication lies in the fact that emergencies occur occasionally, but must be delivered within their deadlines when they occur. The contributions of this work are three-fold: (1) we propose efficient real-time emergency communication protocols based on slot stealing and event-based communication; (2) we build an open-source WirelessHART protocol stack in the Wireless Cyber-Physical Simulator (WCPS) for holistic simulations of wireless control systems; (3) we conduct systematic studies on a coupled water tank system controlled over a 6-hop 21-node WSAN. Our results demonstrate our real-time emergency communication approach enables timely emergency handling, while allowing regular feedback control loops to effectively share resources in WSANs during normal operations. Our work demonstrates the feasibility and efficacy of incorporating emer-

gency alarms into wireless process control systems. Moreover, WCPS 2.0 with the WirelessHART protocol stack and case studies provide an enabling framework for exploring wireless process control design and hence represents a promising step toward practical wireless process control systems.

8. ACKNOWLEDGEMENT

This work was supported, in part, by US NSF through grants 1035773 (CPS), 1320921 (NeTS), 1144552 (NeTS), and NSFC (National Natural Science Foundation of China) through grant 61273038.

9. REFERENCES

[1] `http://www.hartcomm.org`.

[2] `http://wcps.cse.wustl.edu`.

[3] `http://wsn.cse.wustl.edu/index.php/Testbed`.

[4] How does control timing affect performance? analysis and simulation of timing using jitterbug and truetime. *Proceedings of PWC 2003: Personal Wireless Communication, Lecture Notes in Computer Science*, 23(3):16 – 30, June 2003.

[5] P. Afonso, J. Ferreira, and J. Castro. Sensor fault detection and identification in a pilot plant under process control. *Chemical Engineering Research and Design*, 76(4):490–498, 1998.

[6] B. Aminian, J. Araujo, M. Johansson, and K. H. Johansson. Gisoo: a virtual testbed for wireless cyber-physical systems. In *39th Annual Conference of the IEEE Industrial Electronics Society (IECON 2013)*, 2013.

[7] J. Araujo, A. Anta, M. Mazo, J. Faria, A. Hernandez, P. Tabuada, and K. Johansson. Self-triggered control over wireless sensor and actuator networks. In *Distributed Computing in Sensor Systems and Workshops (DCOSS), 2011 International Conference on*, pages 1–9, 2011.

[8] K. J. Aström and R. M. Murray. *Feedback systems: an introduction for scientists and engineers*. Princeton university press, 2010.

[9] G. Baliga, S. Graham, L. Sha, and P. Kumar. Etherware: Domainware for wireless control networks. In *Proceedings of The 7th IEEE International Symposium on Object-oriented Real-time Distributed Computing*, 2004.

[10] J. Chen, S. Kher, and A. Somani. Distributed fault detection of wireless sensor networks. In *Proceedings of the 2006 workshop on Dependability issues in wireless ad hoc networks and sensor networks*, pages 65–72. ACM, 2006.

[11] O. Chipara, C. Wu, C. Lu, and W. Griswold. Interference-Aware Real-Time Flow Scheduling for Wireless Sensor Networks. In *ECRTS'11*.

[12] P. D. Christofides and N. H. El-Farra. *Control of nonlinear and hybrid process systems: Designs for uncertainty, constraints and time-delays*, volume 324. Springer New York, 2005.

[13] B. Demirel, Z. Zou, P. Soldati, and M. Johansson. Modular co-design of controllers and transmission schedules in wirelesshart. In *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*, pages 5951–5958, 2011.

[14] E. Eyisi, J. Bai, D. Riley, J. Weng, Y. Wei, Y. Xue, X. D. Koutsoukos, and J. Sztipanovits. Ncswt: An integrated modeling and simulation tool for networked control systems. In *The 15th International Conference on Hybrid Systems: Computation and Control (HSCC)*, 2012.

[15] S. Han, X. Zhu, A. Mok, D. Chen, and M. Nixon. Reliable and real-time communication in industrial wireless mesh networks. In *Real-Time and Embedded Technology and Applications Symposium (RTAS), 2011 17th IEEE*, 2011.

[16] K. H. Johansson, M. B. Egerstedt, J. Lygeros, and S. S. Sastry. On the Regularization of Zeno Hybrid Automata. *Systems & Control Letters*, 38(3):141–150, 1999.

[17] X. Koutsoukos, N. Kottenstette, J. Hall, P. Antsaklis, and J. Sztipanovits. Passivity-based control design for cyber-physical systems. In *International Workshop on Cyber-Physical Systems-Challenges and Applications*, 2008.

[18] H. Lee, A. Cerpa, and P. Levis. Improving wireless simulation through noise modeling. In *IPSN*, 2007.

[19] B. Li, Z. Sun, K. Mechitov, C. Lu, D. Dyke, G. Agha, and B. Spencer. Realistic case studies of wireless structural control. In *ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS'13)*, April 2013.

[20] B. Li, D. Wang, F. Wang, and Y. Ni. High quality sensor placement for SHM systems: Refocusing on application demands. In *Proc. IEEE INFOCOM'10, San Diego, CA, Mar.*, 2010.

[21] X. Liu, J. Cao, W.-Z. Song, and S. Tang. Distributed sensing for high quality structural health monitoring using wireless sensor networks. In *The 33rd IEEE Real-Time Systems Symposium (RTSS'12)*, 2012.

[22] M. Pajic, S. Sundaram, J. Le Ny, G. J. Pappas, and R. Mangharam. Closing the loop: a simple distributed method for control over wireless networks. In *Proceedings of the 11th international conference on Information Processing in Sensor Networks*, IPSN '12, pages 25–36, New York, NY, USA, 2012. ACM.

[23] I. Rhee, A. Warrier, M. Aia, J. Min, and M. L. Sichitiu. Z-mac: a hybrid mac for wireless sensor networks. *IEEE/ACM Trans. Netw.*, 16(3):511–524, June 2008.

[24] A. Saifullah, C. Wu, P. Tiwari, Y. Xu, Y. Fu, C. Lu, and Y. Chen. Near optimal rate selection for wireless control systems. In *RTAS,12*.

[25] A. Saifullah, Y. Xu, C. Lu, and Y. Chen. Real-time scheduling for WirelessHART networks. In *RTSS*, 2010.

[26] B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, M. Jordan, and S. Sastry. Kalman filtering with intermittent observations. *Automatic Control, IEEE Transactions on*, 49(9):1453–1464, 2004.

[27] Z. Sun, B. Li, D. Dyke, and C. Lu. Evaluation of performances of structural control benchmark problem with time delays from wireless sensor network. In *Joint Conference of the Engineering Mechanics Institute and ASCE Joint Specialty Conference on Probabilistic Mechanics and Structural Reliability (EMI/PMC'12)*, 2012.

[28] Z. Sun, B. Li, S. J. Dyke, and C. Lu. Benchmark problem in active structural control with wireless sensor network. *Structural Control and Health Monitoring*, 2015.

[29] P. Tabuada. Event-triggered real-time scheduling of stabilizing control tasks. *Automatic Control, IEEE Transactions on*, 52(9):1680–1685, 2007.

[30] Y. Wang, S. X. Ding, H. Ye, and G. Wang. A new fault detection scheme for networked control systems subject to uncertain time-varying delay. *Signal Processing, IEEE Transactions on*, 56(10):5258–5268, 2008.

[31] C. Wu, M. Sha, D. Gunatilaka, A. Saifullah, C. Lu, and Y. Chen. Analysis of EDF Scheduling for Wireless Sensor-Actuator Networks. In *IEEE/ACM Symposium on Quality of Service (IWQoS'14)*, May 2014.

[32] S. Zahedi, M. Szczodrak, P. Ji, D. Mylaraswamy, M. Srivastava, and R. Young. Tiered architecture for on-line detection, isolation and repair of faults in wireless sensor networks. In *Military Communications Conference, 2008. MILCOM 2008. IEEE*, pages 1–7. IEEE, 2008.