Closing the Loop: A Simple Distributed Method for Control over Wireless Networks *

Miroslav Pajic
Dept. of Electrical & Systems Eng.
University of Pennsylvania
pajic@seas.upenn.edu

Shreyas Sundaram
Dept. of Electrical & Computer Eng.
University of Waterloo
ssundara@uwaterloo.ca

Jerome Le Ny
Dept. of Electrical & Systems Eng.
University of Pennsylvania
jeromel@seas.upenn.edu

George J. Pappas
Dept. of Electrical & Systems Eng.
University of Pennsylvania
pappasg@seas.upenn.edu

Rahul Mangharam
Dept. of Electrical & Systems Eng.
University of Pennsylvania
rahulm@seas.upenn.edu

ABSTRACT

We present a distributed scheme used for control over a network of wireless nodes. As opposed to traditional networked control schemes where the nodes simply route information to and from a dedicated controller (perhaps performing some encoding along the way), our approach, Wireless Control Network (WCN), treats the network itself as the controller. In other words, the computation of the control law is done in a fully distributed way inside the network. We extend the basic WCN strategy, where at each time-step, each node updates its internal state to be a linear combination of the states of the nodes in its neighborhood. This causes the entire network to behave as a linear dynamical system, with sparsity constraints imposed by the network topology. We demonstrate that with observer style updates, the WCN's robustness to link failures is substantially improved. Furthermore, we show how to design a WCN that can maintain stability even in cases of node failures. We also address the problem of WCN synthesis with guaranteed optimal performance of the plant, with respect to standard cost functions. We extend the synthesis procedure to deal with continuous-time plants and demonstrate how the WCN can be used on a practical, industrial application, using a process-in-the-loop setup with real hardware.

*This work has been partially supported by NSF-CNS 0931239 and NSF-MRI 0923518 grants. It has also been funded by a grant from the Natural Sciences and Engineering Research Council of Canada (NSERC).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
IPSN'12, April 16–20, 2012, Beijing, China.
Copyright 2012 ACM 978-1-4503-1227-1/12/04 ...$10.00.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*; C.3 [**Special-purpose and Application-based Systems**]: *Process control systems*

General Terms

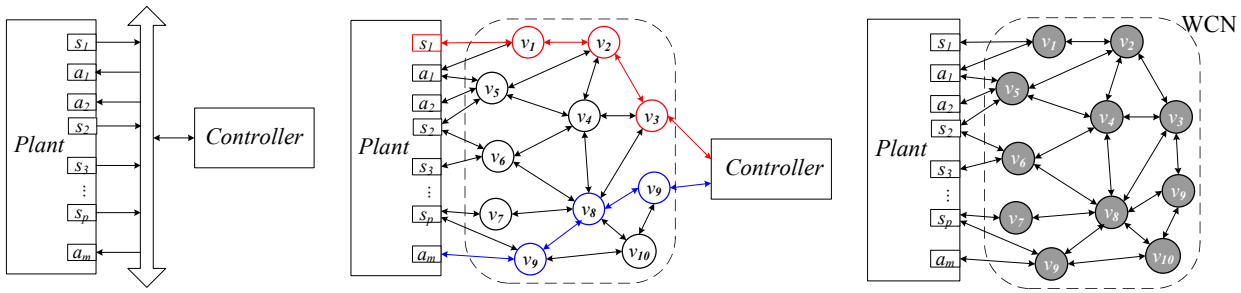
Algorithms, Design, Theory, Performance, Reliability

Keywords

Cooperative control, decentralized control, networked control systems, wireless sensor networks

1. INTRODUCTION

Improvements in the capabilities and cost of wireless technologies have allowed multi-hop wireless networks to be used as a means of (open-loop) monitoring of large-scale industrial plants [1, 2]. With this technology, sensor measurements of plant variables can be transmitted to controllers, data centers and plant operators without the need for excessive wiring, thereby yielding gains in efficiency and flexibility for the operator. However, the use of multi-hop wireless networks in closed-loop feedback control is in its infancy, and is an active area of research [3, 4]. Wireless Networked Control Systems (WNCSs) fundamentally differ from standard wired distributed systems in that the dynamics of the *network* (variable channel capacity, probabilistic connectivity, topological changes, node and link failures) can change the operating points and physical dynamics of the *closed-loop system* [3, 5]. The most important objective of WNCSs is to provide stability of the closed-loop system. An additional requirement is optimality with respect to some appropriate cost function. It is therefore necessary for the network (along with its interfaces to sensors and actuators) to be able to provide some form of guarantee of the control system's stability in the face of the non-idealities of the wireless links and the communication constraints of the wireless network.



(a) Wired Network Control (b) Wireless Network Controlled System (c) Wireless Control Network

Figure 1: Standard architectures for Networked Control Systems; (a) Wired system with a shared bus and dedicated controller; (b) Red links/nodes - routing data from the plant's sensors to the controller; Blue links/nodes - routing data from the controller to the plant's actuators; (c) A multi-hop wireless control network used as a distributed controller.

The most common approach to incorporating WNCSs into the feedback loop is to use it primarily as a communication medium: the nodes in the network simply route information to and from one or more *dedicated controllers*, which are usually specialized CPUs capable of performing computationally expensive procedures (see Fig. 1(b)). The use of dedicated controllers imposes a routing requirement along one or more fixed paths through the network, which must meet the stability constraints, encapsulated by end-to-end delay requirements [6, 7]. However, this assignment of routes is a static setup, which commonly requires global reorganization for changes in the underlying topology, node population and wireless link capacities.

Routing couples the communication, computation and control problems [4, 8, 9]. Therefore, when a new route is required due to topological changes, the computation and control configurations must also be recalculated. Merely inserting a WNCS into the standard network architecture "sensor → channel → controller/estimator → channel → actuator" requires the addition of significant software support [7, 10], as the overhead of completely recomputing the computation and control configurations, due to topological changes or packet drops, is too expensive and does not scale.

1.1 Wireless Control Design Challenge

It should be noted that providing closed-loop stability and performance guarantees for wireless control networks is a challenging problem. On one hand, the control systems community typically abstracts away the systems details and solves the problem for semi-idealized networks with approximated noise distributions and link perturbations [3, 5]. While this approach provides mathematical certainty of the properties of the network, it fails to provide a systematic path to real-world network design. On the other hand, the network systems community uses hardware and software approaches to address open-loop issues, but these fail to provide any guarantees to maintaining stability and performance of closed-loop control. We propose a control scheme over wireless networks that provides closed-loop stability or optimality with respect to standard metrics, while maintaining ease of implementation in real-world networks.

The applications of interest in this work are industrial process control systems (such as natural gas refineries and paper pulp manufacturing plants) and building automation systems. In general, the plant time-constants are on the order of several seconds to a few minutes and the control network is expected to operate at rates of hundreds of milliseconds. While such plants may have as many as 80,000 to 110,000 control loops, they are organized in a hierarchal manner such that networks span 10-20 wireless nodes (per gateway) for low-level control [6]. Therefore, in this work we focus on the networks with up to a few tens of nodes.

Furthermore, the networks might be shared among control loops (i.e., a node may be involved in several feedback loops), and new feedback loops may be added at run-time. Adding new communication loops in a standard WNCS could affect the performance of the existing loops, and the system must be analyzed as a whole. Although techniques have been developed for compositional analysis of WNCS (e.g., [4]), their complexity limits their use in these applications. Therefore, it is necessary to derive a *composable* control scheme, where control loops can be easily added and a simple compositional analysis can be performed at run-time, to ensure that one loop does not affect performance of other loops.

Finally, with the use of asynchronous event triggered network substrates, it is difficult to design, model and analyze control networks and also hard, if not impossible, to verify performance of a system that consists of several event-triggered loops. On the other hand, full network synchronization allows the use of Time-Triggered Architectures (TTAs) where communication and computation are scheduled at particular instances of time (i.e., time slots) [11]. This simplifies modeling of the closed-loop system, which now consists of continuous-time physical dynamics and a communication network. With TTA, the closed-loop system can be modeled as a switched control system [4], allowing the use of existing techniques for switched-system analysis. Therefore, when communication and computation schedules are derived, it is possible to determine if the closed-loop system is stable (i.e., asymptotically stable or mean square stable) for channel errors, with and/or without permanent link or node failures.

1.2 Contributions

In this effort, we build on the work from [12], in which we introduced the Wireless Control Network (WCN), a fundamentally new concept where the network *itself* acts as the controller. We proposed a basic scheme that can be used to guarantee closed-loop system stability. We henceforth refer to this as the "basic WCN". Our contribution

is focused specifically on four major extensions to the basic WCN, making it optimal, more robust and practical:

1.) While the basic WCN in [12] provided closed-loop stability, the key issue of *optimality* was not investigated. We present here a method to extract an optimal WCN configuration, thus providing a greater incentive to adopt WCN in industrial control applications.

2.) The basic WCN was able to provide stable network configurations for a large class of wireless network topologies. However, it was highly susceptible to packet drops greater than 1%. In this work, we present significant robustness improvements, maintaining stability for packet drop rates ***up to 20%*** for a specific network topology and plant. This bridges the gap between the basic WCN and the theoretical upper bound of robustness to packet drops [13].

3.) Furthermore, we propose a method to extract WCN configurations that maintain stability of the closed-loop system even in presence of node failures.

4.) Finally, we illustrate the use of the improved WCN in an industrial process control case study. Using a process-in-the-loop test-bed we demonstrate its ability to optimally control continuous-time physical processes, and to maintain system stability under the presence of node and link failures.

In this paper, we consider scenarios where the network topology is already set, and we present algorithms to configure the WCN to guarantee stability (optimality) for the predefined topology. In [14] we have investigated a dual problem, how to synthesize the network so that a stable WCN configuration exists. The topological conditions from [14], along with the results from this work allow for an integrated decentralized wireless control network design framework.

1.3 Organization of the Paper

The rest of the paper is organized as follows: Sec. 2 describes the concept of the WCN. In Sec. 3 we present a method used to extract optimal WCN configurations. Sec. 4 and 5 extend the WCN scheme to improve its robustness to link and node failures. Sec. 6 describes an approach to employ WCN for control of continuous-time process. Finally, in Sec. 7 we show how the WCN can be used in an industrial, process control application.

1.4 Notation

We use $\mathbf{I}_N$ to denote the $N \times N$ identity matrix, while $\mathbf{I}$ denotes the identity matrix of appropriate dimensions. The notation $\text{diag}(\cdot)$ indicates a square matrix with the quantities inside the brackets on the diagonal, and zeros elsewhere. For a vector $\mathbf{x}$, $\|\mathbf{x}\|$ denotes the Euclidean norm (i.e., $\|\mathbf{x}\| = \sqrt{\mathbf{x}^T\mathbf{x}}$). Finally, $\mathbf{A} \succ \mathbf{0} (\succeq \mathbf{0})$ indicates that matrix $\mathbf{A}$ is positive (semi)definite.

2. WIRELESS CONTROL NETWORKS

2.1 An Intuitive Overview of the WCN

The role of feedback control is to apply inputs to the plant (based on observed outputs) in order to elicit the desired behavior. The exact mapping between observed behavior and applied inputs depends on a mathematical model of the plant, describing how inputs affect the system (over time). Here, we start with a common discrete-time, linear time-invariant model of the form:[1]

$$\begin{aligned} \mathbf{x}[k+1] &= \mathbf{A}\mathbf{x}[k] + \mathbf{B}\mathbf{u}[k] + \mathbf{B_w}\mathbf{u}_w[k] \\ \mathbf{y}[k] &= \mathbf{C}\mathbf{x}[k], \end{aligned} \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{y} \in \mathbb{R}^p$ denote the plant's state and output, $\mathbf{u} \in \mathbb{R}^m$ is the plant's (controllable) input, and $\mathbf{u}_w \in \mathbb{R}^{m_w}$ is the disturbance input.[2] Accordingly, the matrices $\mathbf{A}, \mathbf{B}, \mathbf{B}_w, \mathbf{C}$ have suitable dimensions.

Standard dynamical feedback controllers collect the observed plant outputs $\mathbf{y}[k]$ and generate the control input $\mathbf{u}[k]$ as the output of a linear system of the form:

$$\begin{aligned} \mathbf{x_c}[k+1] &= \mathbf{A_c}\mathbf{x}_c[k] + \mathbf{B_c}\mathbf{y}[k] \\ \mathbf{u}[k] &= \mathbf{C_c}\mathbf{x}_c[k] + \mathbf{D_c}\mathbf{y}[k]. \end{aligned} \tag{2}$$

The vector $\mathbf{x}_c[k]$ denotes the state of the controller, and the matrices $\mathbf{A}_c, \mathbf{B}_c, \mathbf{C}_c$ and $\mathbf{D}_c$ are designed using standard tools from control theory, to ensure that the control inputs are stabilizing. Depending on the control method used, the state of the controller can often be as large as the state of the system itself.

In the above traditional approach to controller design, a wireless network would simply be placed between the controller and the plant to carry information back and forth. The goal of our work is to derive a truly networked and fully distributed control scheme, where the collective computation and communication capabilities of the wireless nodes are fully leveraged to compute the control inputs *in-network*. Intuitively, we propose a simple scheme for each node in the network to follow (using only information from its nearest neighbors at each time-step) that results in the desired network behavior. Essentially, we would like each wireless node to act as a small dynamical controller, with two main differences: (i) the state of the controller at each node will be constrained to be rather small (in order to account for resource and computational constraints), and (ii) in its updates, each node only uses the states of its nearest neighbors (which could include the plant's outputs, if the node is within transmission range of the outputs). Note that the latter condition precludes the need to route information from the plant to each controller in order for it to perform its update. In the rest of this section, we will make these conditions more mathematically precise.

2.2 Model of the Wireless Control Network

To model the WCN we consider the basic WCN setup from Fig. 1(c), where the plant is to be controlled using a multi-hop, fully synchronized wireless network with N nodes. In this paper, we extend the proposed scheme to allow for the design of a WCN that applies inputs in an 'optimal' manner (according to a cost function that we will define later). The plant model is given by (1), where the output vector $\mathbf{y}[k]$ contains the plant's output measurements provided by the sensors $s_1, \ldots, s_p$, while the input vector $\mathbf{u}[k]$ corresponds to the signals applied to the plant by actuators $a_1, \ldots, a_m$. The wireless network is described by a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$ is the set of N nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ represents the radio connectivity (communication topology) in the network (i.e., edge $(v_j, v_i) \in \mathcal{E}$, if node v_i can receive information directly from node v_j).

[1] In Section 6 we will show how continuous-time plants can be cast in this framework using discretization.

[2] We do not have any control over the disturbances.

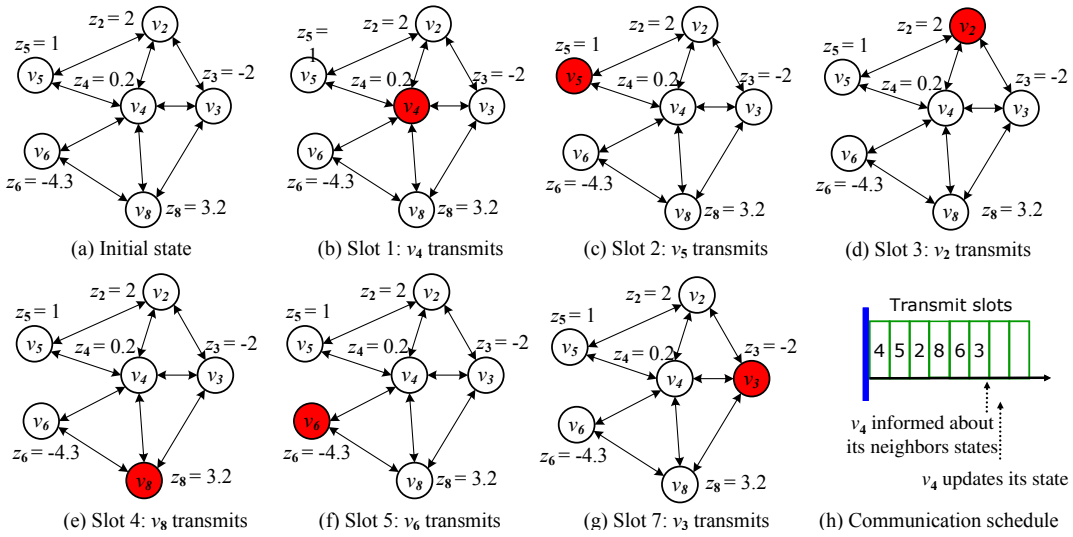


Figure 2: An illustration of the WCN scheme for a simple network.

As mentioned earlier, our scheme views each node v_i as a (small) linear dynamical controller, with (possibly vector) state z_i. Each node updates the state of its controller as a linear combination of the states of its neighbors and its own state. The state update for node v_i can also include a linear combination of the plant outputs from all plant sensors in v_i's neighborhood.

For example, consider the network presented in Fig. 2, where at the beginning of a time frame each node has an initial state value denoted by z_i (Fig. 2(a)). If each node maintains a scalar state, the size of the state is just **2 bytes**.[3] In the first time slot of a frame (Fig. 2(b)) node v_4 transmits its state, and in the second slot node v_5 transmits the state, etc. Finally, in the 6^{th} slot node v_3 is the last node in the frame to transmit its state (Fig. 2(g)). This results in a communication schedule as depicted in Fig. 2(h). After slot 6, node v_4 is informed about all its neighbors' states, which enables it to update its state by activating the WCN task. The task has to compute the updated state value before the node is scheduled for transmission in the next frame.

In the general case, if $z_i[k]$ denotes the i^{th} node's state at time step (i.e., communication frame) k, the **runtime update procedure** is:

$$z_i[k+1] = w_{ii} z_i[k] + \sum_{v_j \in \mathcal{N}_{v_i}} w_{ij} z_j[k] + \sum_{s_j \in \mathcal{N}_{v_i}} h_{ij} y_j[k], \quad (3)$$

where the neighborhood of a vertex v is represented as $\mathcal{N}_v$ and $y_j[k]$ is the measurement provided by sensor s_j. We will model the resource constraints of each node in the network by limiting the size of the state vector that can be maintained by each node.[4] Note the similarity of the update (3) to the state update equation for traditional dynamical controllers of the form (2); the state $z_i[k]$ plays the role of $x_c[k]$, the weights w_{ii} and w_{ij} play the role of $\mathbf{A}_c$ and the columns of $\mathbf{B}_c$, respectively.

To enable interaction between the network and the plant, each actuator a_i applies input $u_i[k]$, which is computed as a linear combination of states from the nodes in the neighborhood of the actuator:

$$u_i[k] = \sum_{j \in \mathcal{N}_{a_i}} g_{ij} z_j[k]. \quad (4)$$

Once again, note the resemblance of this applied input to the input applied by a standard controller of the form (2). Therefore, the behavior of each node in the network is determined by values w_{ij}, h_{ij} and g_{ij}. Aggregating the state values of all nodes at time step k into the value vector $\mathbf{z}[k]$, we see that the above individual controllers at each node collectively cause the entire network to act as a dynamical controller of the form:

$$\mathbf{z}[k+1] = \underbrace{\begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1N} \\ w_{21} & w_{22} & \cdots & w_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ w_{N1} & w_{N2} & \cdots & w_{NN} \end{bmatrix}}_{\mathbf{W}} \mathbf{z}[k] + \underbrace{\begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1p} \\ h_{21} & h_{22} & \cdots & h_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ h_{N1} & h_{N2} & \cdots & h_{Np} \end{bmatrix}}_{\mathbf{H}} \mathbf{y}[k]$$

$$= \mathbf{W}\mathbf{z}[k] + \mathbf{H}\mathbf{y}[k] \ ,$$

$$\mathbf{u}[k] = \underbrace{\begin{bmatrix} g_{11} & g_{12} & \cdots & g_{1N} \\ g_{21} & g_{22} & \cdots & g_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ g_{m1} & g_{m2} & \cdots & g_{mN} \end{bmatrix}}_{\mathbf{G}} \mathbf{z}[k] = \mathbf{G}\mathbf{z}[k]$$

for all $k \in \mathbb{N}$. Since for all $i \in \{1, \ldots, N\}$, $w_{ij} = 0$ if $v_j \notin \mathcal{N}_{v_i}$, $h_{ij} = 0$ if $s_j \notin \mathcal{N}_{v_i}$, and $g_{ij} = 0$ if $v_j \notin \mathcal{N}_{a_i}$ the matrices $\mathbf{W}, \mathbf{H}$ and $\mathbf{G}$ are *structured*, with sparsity constraints determined by the network topology at design time.

[3] Given that standard analog-to-digital converters have a precision of 12-16 bits, two bytes suffice for scalar values.

[4] To present our results, we will focus on the case where each node's state is a scalar. The general case, where each heterogeneous node can maintain a vector state with possibly different dimensions, can be treated with a natural extension of our approach (e.g., see [12]).

Throughout the rest of the paper, we will define Ψ to be the set of all tuples $(\mathbf{W}, \mathbf{H}, \mathbf{G}) \in \mathbb{R}^{N\times N} \times \mathbb{R}^{N\times p} \times \mathbb{R}^{m\times N}$ satisfying the aforementioned sparsity constraints. Denoting the overall system state (plant's state and states of all nodes in the network) by $\hat{\mathbf{x}}[k] = \begin{bmatrix}\mathbf{x}[k]^T & \mathbf{z}[k]^T\end{bmatrix}^T$, the closed-loop system evolves as:

$$\begin{aligned}\hat{\mathbf{x}}[k+1] &= \underbrace{\begin{bmatrix}\mathbf{A} & \mathbf{BG}\\ \mathbf{HC} & \mathbf{W}\end{bmatrix}}_{\hat{\mathbf{A}}}\underbrace{\begin{bmatrix}\mathbf{x}[k]\\ \mathbf{z}[k]\end{bmatrix}}_{\hat{\mathbf{x}}[k]} + \underbrace{\begin{bmatrix}\mathbf{B}_w\\ \mathbf{0}\end{bmatrix}}_{\hat{\mathbf{B}}}\mathbf{u}_w \\ &= \hat{\mathbf{A}}\hat{\mathbf{x}}[k] + \hat{\mathbf{B}}\mathbf{u}_w[k].\end{aligned} \tag{5}$$

To use the WCN runtime scheme it is essential to determine an appropriate set of link weights (w_{ij}, h_{ij} and g_{ij}) at **design-time**, so that the closed loop system is asymptotically stable.[5] When there are no disturbances (i.e., $\mathbf{u}_w[k] \equiv \mathbf{0}$), an initial procedure was proposed for the basic WCN that guarantees that the closed-loop system is stable, or Mean Square Stable (MSS) if the communication links are unreliable.[6]

2.2.1 Advantages of the WCN

The WCN introduces very low communication and computation overhead. The linear iterative runtime procedure (3) is computationally very inexpensive as each node only computes a linear combination of its value and values of its neighbors. This makes it suitable for resource constrained, low-power wireless nodes (e.g., Tmote). Furthermore, the communication overhead is also very small, as each node needs to transmit only its own state once per frame. In the case when a node maintains a scalar state it transmits only *2 bytes* in each message, making it suitable to combine this scheme with periodic message transmissions in existing wireless systems.

Another key benefit is that the WCN can easily handle plants with multiple geographically distributed sensors and actuators, a case that is not easily handled by the "sensor → channel → controller/estimator → channel → actuator" setup commonly adopted in networked control design. The existence of a centralized controller might impose a requirement that the sampling time of the plant is greater than or equal to the sum of communication delays, from sensors to the controller and from the controller to the actuator, along with the time required for the computation of the control algorithm. The WCN does not rely on the existence of centralized controllers, and inherently captures the case of nodes exchanging values with the plant at various points in the network. Therefore, when the WCN is used, the network diameter does not affect the sampling period of the plant.

Finally, the WCN utilizes a simple transmission schedule where each node is active only once during a TDMA cycle and the control-loop does not impose end-to-end delay requirements. This allows the network operator to decouple the computation schedule from the communication schedule, which significantly simplifies closed-loop system design and enables compositional design and analysis. As long as each node can send additional states in a single transmission packet, and schedule computation of additional linear procedures, adding a new control loop will not affect the performance of the existing control loops. For example, consider IEEE 802.14.5 networks that have the maximal packet size of 128 bytes. If each plant is controlled using the WCN scheme where all nodes maintain a scalar 16 bit state value, then up to 64 plants can be controlled in parallel.

In this paper, we provide an enhanced WCN scheme that maintains all of these desirable properties, and further incorporates optimality and robustness metrics into the basic scheme.

2.2.2 Synchronization Requirements

For the network sizes considered here, it is necessary to use either hardware-based out-of-band synchronization or some of the built-in synchronization protocols that guarantee low synchronization error between neighboring nodes (e.g., the approach described in [17] guarantees that the maximal synchronization error between neighboring nodes is less than 1 μs). Even for 10 μs synchronization error between neighboring nodes, for large scale networks with the network diameter less than 100 nodes, maximal synchronization error between nodes is less than 1ms, which is significantly smaller than standard sampling rates of the plant when WCN is used. For example, if communication frames that consist of 16 slots are used, where each slot is 10 ms wide, the sampling period of the plant equals to 160 ms. In this case, synchronization errors would take less than 1% of the sampling period. We employ a synchronized network and use the RT-Link [18] time synchronized protocol in our evaluation. Time synchronized network protocols are the norm in the control automation industry, and two recent standards, WirelessHART[1] and ISA 100.11a [19] utilize a time division multiplexing link protocol.

3. SYNTHESIS OF AN OPTIMAL WCN

In this section we present a design-time method to determine a WCN configuration (i.e., link weights for a network with predefined topology) that minimizes effects of the disturbances acting on the system. More specifically, consider the model of the closed-loop system from (5), and assume that we want to minimize the influence of the disturbance input $\mathbf{u}_w$ on the vector $\hat{\mathbf{y}} = \hat{\mathbf{C}}\hat{\mathbf{x}}[k]$, for some matrix $\hat{\mathbf{C}}$. For example, if we would like to focus on minimizing the effects on the plant's state $\mathbf{x}$, we would define $\hat{\mathbf{C}} = \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}$. Thus, we can consider the vector $\hat{\mathbf{y}}$ as the 'output' of the system:

$$\begin{aligned}\hat{\mathbf{x}}[k+1] &= \hat{\mathbf{A}}\hat{\mathbf{x}}[k] + \hat{\mathbf{B}}\mathbf{u}_w[k] \\ \hat{\mathbf{y}} &= \hat{\mathbf{C}}\hat{\mathbf{x}}[k].\end{aligned} \tag{6}$$

To determine the effect of the disturbance on the system's outputs, it is necessary to define a unit of measure to capture the 'size' of discrete-time signals. We will use the norms: $\|\mathbf{v}\|_{\ell_2} \triangleq \left(\sum_{k=0}^{\infty}\|\mathbf{v}[k]\|^2\right)^{1/2}$ and $\|\mathbf{v}\|_{\ell_\infty} \triangleq \sup_{k\geq 0}\|\mathbf{v}[k]\|$. Furthermore, the notion of a *system gain* is introduced to classify the worst-case system response to limited energy input disturbances.

DEFINITION 1 ([20]). *System gains for the discrete-time system* (6) *are defined as:*

[5] A linear system $\mathbf{x}[k+1] = \mathbf{A}\mathbf{x}[k]$ is asymptotically stable if for any $\mathbf{x}[0]$, $\lim_{k\to\infty}\mathbf{x}[k] = \mathbf{0}$. This is equivalent to saying that all eigenvalues of $\mathbf{A}$ have magnitude less than 1.

[6] The system $\mathbf{x}[k+1] = \mathbf{A}_{\theta(\mathbf{k})}\mathbf{x}[k]$, where subscript $\theta(k)$ describes time-variations caused by (probabilistic) drops of communication packets, is mean-square stable if for any initial state $(\mathbf{x}[0], \theta(0))$, $\lim_{k\to\infty}\mathbb{E}\left[\|\mathbf{x}[k]\|^2\right] = 0$, where the expectation is with respect to the probability distribution of the packet drop sequence $\theta(k)$ [15, 16].

- ***Energy-to-Peak Gain:*** $\gamma_{ep} = \sup_{\|\mathbf{u}_w\|_{\ell_2} \leq 1} \|\hat{\mathbf{y}}\|_{\ell_\infty}$
- ***Energy-to-Energy Gain:*** $\gamma_{ee} = \sup_{\|\mathbf{u}_w\|_{\ell_2} \leq 1} \|\hat{\mathbf{y}}\|_{\ell_2}$

We will require the following result from [21].

THEOREM 1. *Suppose that the system* (6) *is asymptotically stable and consider any nonnegative* $\gamma \in \mathbb{R}$.
(a) $\gamma_{ep} < \gamma$ *if and only if there exist matrices* $\mathcal{X} \succ 0, \Upsilon \succeq 0$ *and* $\mathcal{Z}$ *such that* $\Upsilon \prec \gamma \mathbf{I}$ *and*

$$\mathcal{R}(\mathcal{X}, \mathcal{Z}, \Upsilon, \mathcal{X}^{-1}) = \begin{bmatrix} \mathcal{X} & \mathcal{Z} & \hat{\mathbf{A}} & \hat{\mathbf{B}} \\ \mathcal{Z}^T & \Upsilon & \hat{\mathbf{C}} & \mathbf{0} \\ \hat{\mathbf{A}}^T & \hat{\mathbf{C}}^T & \mathcal{X}^{-1} & \mathbf{0} \\ \hat{\mathbf{B}}^T & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \succ 0 \quad (7)$$

(b) $\gamma_{ee} < \gamma$ *if and only if there exists matrices* $\mathcal{X} \succ 0, \Upsilon \succeq 0$ *such that* $\Upsilon \prec \gamma^2 \mathbf{I}$ *and* (7) *holds for* $\mathcal{Z} = \mathbf{0}$.

Only the matrix $\hat{\mathbf{A}}$ contains the WCN parameters, aggregated in the structured matrices $\mathbf{W}, \mathbf{G}, \mathbf{H}$ (from (5)). Our goal is to determine matrices $\mathbf{W}, \mathbf{G}, \mathbf{H}$ that satisfy the imposed structural constraints, along with matrices $\mathcal{X}, \mathcal{Z}, \Upsilon$, for which the value γ is minimized.

The constraint (7) is linear with respect to all variables, except the matrix $\mathcal{X}$ (due to the presence of the term $\mathcal{X}^{-1}$). This term causes the problem of solving the matrix inequality to be non-convex. To ameliorate this issue and efficiently solve the optimization problem, we linearize the $\mathcal{X}^{-1}$ term. As shown in [21], the Taylor series expansion of $\mathcal{X}^{-1}$ 'around' any matrix $\mathcal{X}_k$ is

$$LIN(\mathcal{X}^{-1}, \mathcal{X}_k) = \mathcal{X}_k^{-1} - \mathcal{X}_k^{-1}(\mathcal{X} - \mathcal{X}_k)\mathcal{X}_k^{-1}. \quad (8)$$

With the above linearization we obtain a linear matrix inequality (LMI) for the constraint 7. As in [21, 22], we can now define an iterative algorithm to minimize γ, while ensuring that the constraint from (7) is satisfied. This is achieved by replacing the term $\mathcal{X}^{-1}$ with $LIN(\mathcal{X}^{-1}, \mathcal{X}_k)$ in each iteration, which results in Algorithm 1. Note that $\hat{\mathbf{A}}(\mathbf{W}, \mathbf{H}, \mathbf{G})$ denotes the matrix $\hat{\mathbf{A}}$ obtained from matrices $\mathbf{W}$, $\mathbf{H}$, $\mathbf{G}$ as defined in (5). Finally, for γ obtained from Algorithm 1, $\sqrt{\gamma}$ should be used if we had optimized for γ_{ee}.

Consider the sequence $\{\gamma_k\}_{k \geq 0}$ obtained from Algorithm 1. As shown in [21], the linearization from (8) guarantees that for each $k \geq 0$, in step $k+1$ there exists a feasible matrix in an open neighborhood of the point $\mathcal{X}_k$ for which there exists γ, such that $\gamma \leq \gamma_k$. Since γ_{k+1} is the minimum in that iteration, it follows that $\gamma_{k+1} \leq \gamma$. Thus, the sequence $\{\gamma_k\}_{k \geq 0}$ is non-increasing and bounded ($\gamma_k \geq 0$), meaning that it will always converge. Since we are optimizing a convex function over a non-convex set, by linearizing the constraints we might obtain a sub-optimal WCN configuration. The final result and the convergence rate depend on the initial point (from Step 1. of the algorithm). Finally, the smallest ϵ for which we can find an optimal controller can be obtained using bisection on the parameter ϵ.

Algorithm 1 ***Design-time*** procedure used to extract optimal WCN configuration

1. Set $\epsilon > 0, k = 0$. Find a feasible point $\mathcal{X}_0, \mathcal{Y}_0, \Upsilon_0 \succ 0$, $\hat{\mathbf{A}}(\mathbf{W_0}, \mathbf{H_0}, \mathbf{G_0})$, such that $\mathcal{R}(\mathcal{X}_0, \mathcal{Z}, \Upsilon_0, \mathcal{Y}_0) \succ 0$, $\mathcal{X}_0 \succeq \mathcal{Y}_0^{-1}$ and $(\mathbf{W}_0, \mathbf{H}_0, \mathbf{G}_0) \in \Psi$. If a feasible point does not exist, it is not possible to stabilize the system with this network topology.

2. At iteration k ($k \geq 0$), from $\mathcal{X}_k$ obtain the matrix $\mathcal{X}_{k+1}$ and scalar γ_{k+1} by solving the LMI problem

$$\mathcal{X}_{k+1} = \arg \min_{\mathcal{X}, \mathcal{Z}, \Upsilon, \mathbf{W}, \mathbf{H}, \mathbf{G}, \gamma_{k+1}} \gamma_{k+1} \quad (9)$$

$$\mathcal{R}(\mathcal{X}, \mathcal{Z}, \Upsilon, LIN(\mathcal{X}^{-1}, \mathcal{X}_k)) \succ 0, \quad (10)$$

$$\Upsilon \prec \gamma_{k+1} \mathbf{I}, \quad (11)$$

$$(\mathbf{W}, \mathbf{H}, \mathbf{G}) \in \Psi, \ \ \mathcal{X} \succ 0, \Upsilon \succeq 0 \quad (12)$$

if γ_{ee} is being optimized, add the constraint $\mathcal{Z} = \mathbf{0}$.

3. If $\gamma_{k+1} < \epsilon$ stop the algorithm. Otherwise, set $k = k+1$ and go to the step **2**.

4. ROBUSTNESS TO LINK FAILURES

We now describe the main limitation of the basic WCN, and extend the WCN scheme to improve its robustness to link failures.

The unreliability of wireless communication links is one of the main drawbacks when wireless networks are used for control. When communication links in the feedback loop fail according to a given probability distribution, the notion of asymptotic stability is typically relaxed to settle for *mean square stability* (MSS), where the expected value of the norm of the state stays bounded. For the basic WCN, we proposed a design-time procedure that can be used to extract a stabilizing configuration that guarantees MSS despite unreliable communication links [12]. For example, consider the system from Fig. 3 with a scalar plant, where $\alpha = 2$ (the plant is unstable), and assume that the link between node v_2 and the actuator is reliable (i.e., never drops packets). The basic WCN scheme, where each node maintains a scalar state, guarantees that the closed-loop system is MSS for probabilities of packet drops $\leq 1.18\%$.

To place this result in context, it is worth comparing it with the theoretical limit of robustness in lossy networks from [13]. The work in [13] considers a system with a plant controlled by a centralized controller, which is connected to the plant using a single wireless link between a sensor and the controller. In addition, the controller is connected to the actuators with a set of wired connections. It was shown that for this setup, the system can not be stabilized with a linear controller for probability of message drops p greater than $\frac{1}{|\lambda_{max}|^2}$, where $|\lambda_{max}|$ denotes the maximal norm of the plant's eigenvalues (i.e., eigenvalues of $\mathbf{A}$ from (1)). For the plant from Fig. 3, this would mean that a centralized controller in the aforementioned setup cannot provide MSS of the plant if the probability of message drops is higher than 25% (since $\alpha = 2$). This value is significantly larger than the 1.18% value obtained when the basic WCN scheme is used. We now show how the basic WCN formulation presented in (3), (4) can be modified to significantly improve tolerance to packet drops.

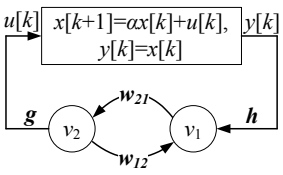


Figure 3: An example of the WCN: A plant with a scalar state controlled by a WCN.

4.1 WCN with Observer Style Updates

To improve WCN robustness to independent link failures, we now allow each node in the network to use different weights in each time step, depending on which neighbors' transmissions were successfully received. Thus, we define the update procedure as:

$$z_j[k+1] = \tilde{w}_{jj} z_j[k] + \sum_{i \in \mathcal{N}_{v_j}} \tilde{w}_{ji} z_i[k], \text{[7]} \qquad (13)$$

where $\tilde{w}_{ji} = 0$ if the message from the node v_i was not received, or w_{ji} otherwise.[8] More importantly, $\tilde{w}_{jj}$ depends on a newly introduced set of link weights (q_{ji}): $\tilde{w}_{jj} = w_{jj} - \sum_{i \in \mathcal{N}_{v_j}} \tilde{q}_{ji}$. Here, $\tilde{q}_{ji} = 0$ if the message from the node v_i was not received, and q_{ji} (a free parameter that will be carefully designed) otherwise.

To model the WCN that employs the above scheme, we need to model the links in the network. We utilize the approach proposed in [16], where each unreliable link $\xi_{ji} = (v_i, v_j)$ (i.e., $v_i \to v_j$) can be modeled as a memoryless, discrete, independent and identically distributed (IID) random process ξ_{ji}. Here, IID implies that the random variables $\{\xi_{ji}[k]\}_{k \geq 0}$ are IID.[9] For each link, these random processes map each transmitted value t_{ji} into a received value $\xi_{ji}[k] t_{ji}$ (see Fig. 4).

With this link model, (13) can be described as:

$$z_j[k+1] = (w_{jj} - \sum_{i \in \mathcal{N}_{v_j}} \xi_{ji} q_{ji}) z_j[k] + \sum_{i \in \mathcal{N}_{v_j}} \xi_{ji} w_{ji} z_i[k],$$

REMARK 1. *If we consider the case with reliable communication links, the update procedure for each node v_j in the network can be described as:*

$$z_j[k+1] = w_{jj} z_j[k] + \sum_{i \in \mathcal{N}_{v_j}} (w_{ji} z_i[k] - q_{ji} z_j[k]), \qquad (14)$$

Since the above equation has the standard observer structure [23], we refer to this scheme as the WCN with observer style updates (as in [24]).

Following the approach from [16], each link described with a random process ξ_{ji} can be specified with a fixed gain, corresponding to the mean value of the random variable, and the zero-mean random part: $\xi_{ji} = \mu_{ji} + \Delta_{ji}$. For example, if each link (i.e., random process ξ_{ji}) is described as a Bernoulli process with probability $p_{ji} \leq 1$ (i.e., the link delivers the transmitted message with probability p_{ji}), then $\mu_{ji} = p_{ji}$ and Δ_{ji} can have values $-p_{ji}$ and $1 - p_{ji}$, with probabilities $1 - p_{ji}$ and p_{ji}, respectively. Therefore, the above procedure becomes:

$$z_j[k+1] = (w_{jj} - \sum_{i \in \mathcal{N}_{v_j}} \mu_{ji} q_{ji}) z_j[k] + \sum_{i \in \mathcal{N}_{v_j}} \mu_{ji} w_{ji} z_i[k] + \sum_{i \in \mathcal{N}_{v_j}} \Delta_{ji} (w_{ji} z_i[k] - q_{ji} z_j[k]).$$

We define $r_t[k] := (w_{ji} z_i[k] - q_{ji} z_j[k])$, for each link $t = (v_i, v_j)$. Also, for each link $t = (s_i, v_j)$ we denote $r_t[k] := (h_{ji} y_i[k] - q_{ji} z_j[k])$. After aggregating all of the $r_t[k]$'s in a vector $\mathbf{r}[k]$ of length N_l (where N_l is the number of links), we obtain:

$$\mathbf{r}[k] = \mathbf{J}^{or} \begin{bmatrix} \mathbf{y}[k] \\ \mathbf{z}[k] \end{bmatrix} = \underbrace{\mathbf{J}^{or} \begin{bmatrix} \mathbf{C} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_N \end{bmatrix}}_{\hat{\mathbf{J}}^{or}} \hat{\mathbf{x}}[k]. \qquad (15)$$

Each row of the matrix $\mathbf{J}^{or} \in \mathbb{R}^{N_l \times (N+p)}$ contains up to two nonzero elements, equal to a gain w_t, h_t, g_t or $-q_t$.

This allows us to model the behavior of the closed-loop system with unreliable communication. Specifically, the update equation for each node v_j is:

$$z_j[k+1] = (w_{jj} - \sum_{i \in \mathcal{N}_{v_j}} \mu_{ji} q_{ji}) z_j[k] + \sum_{t=(v_i,v_j)} \mu_t w_t z_i[k] + \sum_{t=(s_i,v_j)} \mu_t h_t y_i[k] + \sum_{t=(v_i,v_j)} \Delta_t[k] r_t[k] + \sum_{t=(s_i,v_j)} \Delta_t[k] r_t[k]$$

Similarly, the input value applied by each actuator at time k is:

$$u_j[k] = \sum_{t=(v_i,a_j)} \mu_t g_t z_i[k] + \sum_{t=(v_i,a_j)} \Delta_t[k] r_t[k].$$

Finally, denoting $\Delta[k] = \text{diag}(\{\Delta_t[k]\}_{t=1}^{N_l})$, the above expressions can be written in vector form as:

$$\mathbf{z}[k+1] = \mathbf{W}_\mu \mathbf{z}[k] + \mathbf{H}_\mu \mathbf{y}[k] + \mathbf{J}_v^{dst} \Delta[k] \mathbf{r}[k], \qquad (16)$$

$$\mathbf{u}[k] = \mathbf{G}_\mu \mathbf{z}[k] + \mathbf{J}_u^{dst} \Delta[k] \mathbf{r}[k], \qquad (17)$$

where all elements of matrices $\mathbf{W}_\mu, \mathbf{H}_\mu$ and $\mathbf{G}_\mu$ (except the diagonal entries of $\mathbf{W}_\mu$) are of the form $\mu_{ji} w_{ji}, \mu_{ji} h_{ji}$ and $\mu_{ji} g_{ji}$, respectively. The diagonal entries of $\mathbf{W}_\mu$ are of the form $w_{jj} - \sum_{i \in \mathcal{N}_{v_j}} \mu_{ji} q_{ji}$. The binary matrices $\mathbf{J}_v^{dst}$ and $\mathbf{J}_u^{dst}$ are designed in a way that each row of the matrices selects elements of the vector $\Delta[k]\mathbf{r}[k]$ that are added to the linear combinations calculated by the nodes and the actuators. If we denote $\mathbf{J}^{dst} = \begin{bmatrix} \mathbf{J}_u^{dst} \\ \mathbf{J}_v^{dst} \end{bmatrix}$ the overall system with unreliable links can be modeled as:

$$\hat{\mathbf{x}}[k+1] = \underbrace{\begin{bmatrix} \mathbf{A} & \mathbf{B}\mathbf{G}_\mu \\ \mathbf{H}_\mu \mathbf{C} & \mathbf{W}_\mu \end{bmatrix}}_{\hat{\mathbf{A}}_\mu} \hat{\mathbf{x}}[k] + \underbrace{\begin{bmatrix} \mathbf{B} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_N \end{bmatrix} \mathbf{J}^{dst}}_{\hat{\mathbf{J}}^{dst}} \mathbf{\Delta}[k] \mathbf{r}[k], \qquad (18)$$

with $\mathbf{r}[k]$ given by (15). Now, using the same approach as in [16, 12], the following theorem can be proven.

THEOREM 2. *The system from* (18) *is MSS if and only if exist matrices $\mathcal{X}, \mathcal{Y} \succ 0$ and scalars $\alpha_1, ..., \alpha_{N_l}$ such that*

$$\begin{bmatrix} \mathcal{X} - \hat{\mathbf{J}}^{dst} diag\{\alpha\} (\hat{\mathbf{J}}^{dst})^T & \hat{\mathbf{A}}_\mu \\ \hat{\mathbf{A}}_\mu^T & \mathcal{Y} \end{bmatrix} \succ 0 \qquad (19)$$

$$\mathcal{Y} = \mathcal{X}^{-1} \qquad (20)$$

$$\alpha_i \geq \sigma_i^2 (\hat{\mathbf{J}}^{or})_i \mathcal{Y}^{-1} (\hat{\mathbf{J}}^{or})_i^T, \ \forall i \in \{1, \ldots, N_l\} \qquad (21)$$

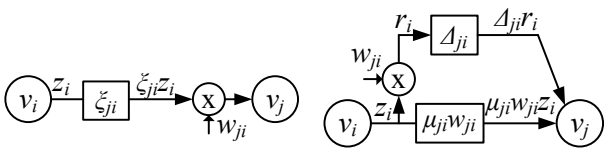


Figure 4: Communication over a non-deterministic channel; (a) A link between nodes v_i and v_j; (b) Link transformation into a robust control form.

[7] A similar update is introduced for nodes that receive sensor values. This part has been omitted for ease of exposition.

[8] Although these weights are technically time varying (i.e., they depend on k), we use this notation for simplicity.

[9] We will address these assumptions later in this section.

where $(\hat{\mathbf{J}}^{or})_i$ *denotes the* i^{th} *row of the matrix* $\hat{\mathbf{J}}^{or}$.

A procedure based on LMIs, with the same structure as Algorithm 1, can be used in this case to compute a WCN configuration that guarantees MSS of the closed-loop system with error-prone links. The difference from Algorithm 1 is that in Step 2, the following problem should be solved:

$$\mathcal{X}_{k+1} = \arg \min_{\mathcal{X},\mathcal{Y},\Upsilon,\mathbf{W},\mathbf{H},\mathbf{G}} tr(\Upsilon)$$
$$\mathcal{Y} - LIN(\mathcal{X}^{-1}, \mathcal{X}_k) \prec \Upsilon, \quad \mathcal{X} \succeq \mathcal{Y}^{-1}$$
such that the constraints from (19),(21),(12) are valid,

where $tr(\mathbf{A})$ denotes the trace of the matrix $\mathbf{A}$. Note that the above algorithm adds only one additional LMI constraint for each link in the network.

4.1.1 Validity of the Assumptions

While developing the model of the WCN from (16), we have assumed that all links in the network are memoryless and independent. Memoryless channels can be obtained if channel hopping is used at the network layer [25]. However, the physical placement of the nodes might introduce correlation between some of the network links.

If these IID assumptions are not valid (or too simplistic), we must model correlation between links along with more complex link failures (such as those induced by a Markov process). In these cases, an approach similar to [15] can be used, which would result in an exponential number of additional constraints introduced to deal with link failures (compared to the linear number of additional constraints introduced under the IID assumption of independent and memoryless channels). Except for very large scale systems, the observer style update procedure is practical as the computation of WCN configurations $(\mathbf{W}, \mathbf{H}, \mathbf{G})$ is only required at design time.

4.1.2 Evaluation

We evaluated the performance of the proposed scheme by modeling all links as independent Bernoulli processes. To analyze robustness of the WCN with observer style updates, we first analyzed the performance of WCNs with $N \geq 2$ nodes that create a complete graph. The WCN is used for control of a single-state plant shown in Fig. 3 (with $\alpha > 1$). Node v_1 receives the plant output $y[k] = x[k]$ at each time-step k, and the input to the plant is derived as a scaled version of the transmission of the node v_2 (i.e., $u[k] = gz_2[k]$ for a scalar g). Using the bisection method from [15], we extracted the maximal probabilities of message drops (p_m) for which there exists a stabilizing configuration that ensures MSS.

We considered two scenarios: In the first scenario, we have compared the performance of the basic WCN with that of the WCN with observer style updates (denoted oWCN). We analyzed networks where all the links are unreliable, described with the same probability of packet drops p (including the links between the plant and the network nodes). The results are presented in Fig. 5(a). In addition, we have investigated the case where the link between node v_2 and the plant's actuator is reliable (without any packet drops). The results are shown in Fig. 5(b). As can be observed, the proposed scheme **significantly** improves system robustness to link failures. *For example, the WCN with observer style updates guarantees MSS for the system from Fig. 3 even when the probability of link failures is more than 20% (compared to 1.5% for the basic WCN).* Similarly, going back to the discussion from the beginning of the section, we have shown in this simple example that the WCN performance is much closer to that of the optimal centralized controllers used for control over wireless links (guaranteeing MSS with up to 25% packet drops).

	WCN (scalar state)	WCN ($\mathbb{R}^2$ state)	$oWCN$ (scalar state)	$oWCN$ ($\mathbb{R}^2$ state)
$N=2$	$p_m = 0.69\%$	$p_m = 0.72\%$	$p_m = 1.64\%$	$p_m = 1.82\%$
$N=3$	$p_m = 0.74\%$	$p_m = 0.77\%$	$p_m = 1.66\%$	$p_m = 1.88\%$
$N=4$	$p_m = 0.77\%$	$p_m = 0.79\%$	$p_m = 1.66\%$	$p_m = 1.88\%$

(a) With all links being unreliable

	WCN (scalar state)	WCN ($\mathbb{R}^2$ state)	$oWCN$ (scalar state)	$oWCN$ ($\mathbb{R}^2$ state)
$N=2$	$p_m = 1.18\%$	$p_m = 1.30\%$	$p_m = 10.46\%$	$p_m = 17.82\%$
$N=3$	$p_m = 1.32\%$	$p_m = 1.46\%$	$p_m = 11.24\%$	$p_m = 17.88\%$
$N=4$	$p_m = 1.41\%$	$p_m = 1.54\%$	$p_m = 11.46\%$	$p_m = 17.88\%$
		$oWCN$ ($\mathbb{R}^3$ state)	$oWCN$ ($\mathbb{R}^4$ state)	$oWCN$ ($\mathbb{R}^5$ state)
$N=2$		$p_m = 20.40\%$	$p_m = 20.48\%$	$p_m = 20.64\%$

(b) With a reliable link between the node v_2 and actuator

Figure 5: Maximal probabilities of link failures for which the closed-loop system from Fig. 3 ($\alpha = 2$) is MSS, when controlled without (WCN) and with observer style updates (oWCN).

Using the observer style updates, similar significantly improved results were obtained for the more complex examples from [12], including larger plants with multiple inputs and outputs, controlled by a mesh network with 9 nodes.

5. ROBUSTNESS TO NODE FAILURES

The stability of the closed-loop system, described by (5), can be affected by node crash failures (i.e., nodes that stop working and drop out of the network). Currently, we have considered two approaches to deal with the node failures. One obvious method to deal with up to k node failures is to precompute at the design-time a set of $N_k = \sum_{j=0}^{k} \binom{N}{j}$ different stabilizing configurations $(\mathbf{W}, \mathbf{H}, \mathbf{G})$ that correspond to all possible choices of k or fewer failed nodes. In this case, each node would need to maintain N_k different sets of link weights for all its incoming links. For example, if each node in the WCN maintains a scalar state, a node with d neighbors would have to maintain on the order of $d \cdot N_k$ different scalar weights. The switching between the precomputed stabilizing configurations could be done either by implementing the detection algorithm from [26], or by having the neighbors of failed nodes broadcast the news of the failures throughout the network, which will prompt all nodes to switch to the appropriate choice of $(\mathbf{W}, \mathbf{H}, \mathbf{G})$.

A more sophisticated method for dealing with the node failures would be to design the WCN in a way that even if some of the nodes fail, the closed-loop system remains stable. For simplicity, consider a WCN that can deal with a single node failure. Let us denote with $\hat{\mathbf{A}}_i$ the matrix $\hat{\mathbf{A}}$ from (5) in the case when node i dies. This is equivalent to setting to zero the i^{th} row of matrices $\mathbf{W}$ and $\mathbf{H}$, along with the i^{th} column of $\mathbf{W}$ and $\mathbf{G}$:

$$\hat{\mathbf{A}}_i \triangleq \begin{bmatrix} \mathbf{A} & \mathbf{B}\mathbf{G}\mathbf{I}_N^i \\ \mathbf{I}_N^i\mathbf{H}\mathbf{C} & \mathbf{I}_N^i\mathbf{W}\mathbf{I}_N^i \end{bmatrix}, \; i = 1, \ldots, N, \qquad (22)$$

Here, $\mathbf{I}_N^i$ denotes $N \times N$ diagonal matrix, with all ones on the diagonal except at the i^{th} position. A sufficient condition for system stability in this case is that there exists a positive definite matrix $\mathcal{X}$ (and, thus, a common Lyapunov function $V(\hat{\mathbf{x}}) = \hat{\mathbf{x}}^T \mathcal{X} \hat{\mathbf{x}}$) such that $\mathcal{X} - \hat{\mathbf{A}}^T \mathcal{X} \hat{\mathbf{A}} \succ \mathbf{0}$ and

$$\mathcal{X} - \hat{\mathbf{A}}_i^T \mathcal{X} \hat{\mathbf{A}}_i \succ \mathbf{0}, \ i = 1, 2, \ldots N. \tag{23}$$

Therefore, the procedure from the previous section with additional N LMI constraints, can be used to extract a stabilizing configuration that can deal with a single node failure. However, in this case it is necessary to design the network in a way that guarantees that such a stabilizing configuration exists. Initial results on these topological conditions have been presented in [14].

6. CONTROL OF CONTINUOUS-TIME PLANTS

Optimal and stabilizing WCN configurations can be obtained using algorithms developed from the closed-loop system model (5) that contains a discrete-time model of the plant (1). However, a similar framework can be used for control of continuous-time plants by discretizing the controlled plant, while taking into account a subtle delay introduced by the communication schedule. To illustrate this, consider a standard continuous-time plant model:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}_c \mathbf{x}(t) + \mathbf{B}_c \mathbf{u}(t) \\ \mathbf{y}(t) &= \mathbf{C}_c \mathbf{x}(t), \end{aligned} \tag{24}$$

with input $\mathbf{x}(t) \in \mathbb{R}^n$, output $\mathbf{y}(t) \in \mathbb{R}^p$, $\mathbf{u}(t) \in \mathbb{R}^m$ and matrices $\mathbf{A}_c, \mathbf{B}_c, \mathbf{C}_c$ of the appropriate dimensions.[10] We denote the sampling period of the plant by T, and we assume that all sensors sample the plant outputs at the beginning of the zero-th slot (as shown in Fig. 6(a)). We also assume that all actuators are scheduled to apply their newly calculated inputs at the beginning of the h^{th} time slot. Note that $h > 0$, because from (4) each actuator has to first receive state values from all of its neighbors, before calculating its next plant input. Similarly, from (4) $h \geq \max(d_{a_i})$, where d_{a_i} denotes the number of neighbors of the actuator a_i.

Therefore, the new inputs will be applied to the plant with the delay $\tau = hT_{sl}$, where T_{sl} is the size of communication slots. This results in the input signal with the form shown in Fig. 6(b). Denoting the number of slots in a communication frame by F, we can write $T = FT_{sl}$. Using the approach from [3, 5], we describe the system:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}_c \mathbf{x}(t) + \mathbf{B}_c \mathbf{u}(t), \\ \mathbf{y}(t) &= \mathbf{C}_c \mathbf{x}(t), \quad t \in [kT + \tau, (k+1)T + \tau), \\ \mathbf{u}(t^+) &= \mathbf{G}\mathbf{z}[k], \quad t \in \{kT + \tau, k = 0, 1, 2, \ldots\} \end{aligned} \tag{25}$$

where $\mathbf{u}(t^+)$ is a piecewise continuous function and only changes values at time instances $kT + \tau, \ k = 0, 1, \ldots$. From the above equation, the discretized model of the system with the sampling period T can be represented as [23]:

$$\begin{aligned} \mathbf{x}[k+1] &= \mathbf{A}\mathbf{x}[k] + \mathbf{B}\mathbf{G}\mathbf{z}[k] + \mathbf{B}^-\mathbf{G}\mathbf{z}[k-1] \\ \mathbf{y}[k] &= \mathbf{C}\mathbf{x}[k], \end{aligned} \tag{26}$$

[10]For simplicity we do not model disturbance inputs to the plant. However, the approach presented in this section can readily handle that scenario.

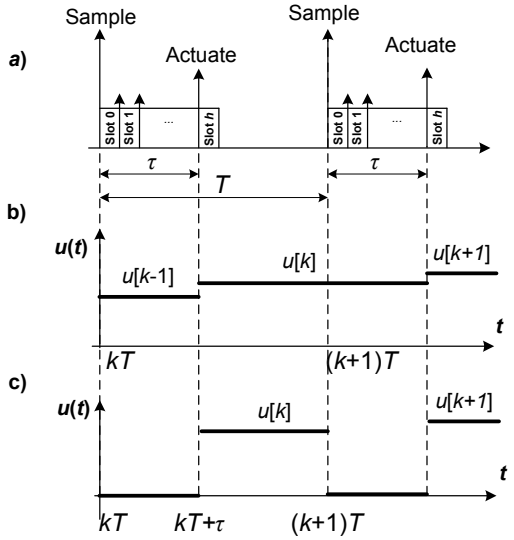


Figure 6: (a) Scheduling sampling/actuation at the start of the slots; (b) Timing diagram for the first type of plant inputs; (c) Plant inputs when actuators reset the inputs at the beginning of the frames.

where $\mathbf{x}[k] = \mathbf{x}(kT), \ k \geq 0$ and

$$\mathbf{A} = e^{\mathbf{A}_c T}, \ \mathbf{B} = \int_0^{T-\tau} e^{\mathbf{A}_c \delta} \mathbf{B}_c d\delta, \ \mathbf{B}^- = \int_{T-\tau}^{T} e^{\mathbf{A}_c \delta} \mathbf{B}_c d\delta. \tag{27}$$

When the communication schedule is extracted and the network is configured, the matrices $\mathbf{A}, \mathbf{B}$ and $\mathbf{B}^-$ obtain fixed-values that depend on the continuous-time plant dynamics, communication frame size T (i.e., the sampling period of the plant) and the utilized communication schedule (as it determines the value for h).

If each actuator applies its current input only until the end of the corresponding frame and then forces its input to zero until the next actuation slot (i.e., h^{th} slot), the input signals would have the form shown in Fig. 6(c) (instead of the form from Fig. 6(b)). In this case, the discretized system could be specified as in (26), (27), with the difference that $\mathbf{B}^- = \mathbf{0}$. Therefore, the discrete-time system takes the form from (1), and stabilizing and optimal configurations can be obtained using the procedures described in the previous sections. However, due to the delay τ, the resulting discrete-time system could be uncontrollable, which in the general case would mean that there is no stabilizing configuration for the closed-loop system.

In situations where $(\mathbf{A}, \mathbf{B})$ is not controllable it is necessary for all actuators to apply their 'old' inputs until new inputs are available (as shown in Fig. 6(b)). This results in a discrete-time plant that does not have the form from (1), and the previous algorithms cannot be directly employed. However, by defining a new vector $\tilde{\mathbf{x}}[k] \triangleq \begin{bmatrix} \mathbf{x}[k]^T & \mathbf{u}[k-1]^T \end{bmatrix}^T$ the discrete-time system can be described as:

$$\begin{aligned} \tilde{\mathbf{x}}[k+1] &= \underbrace{\begin{bmatrix} \mathbf{A} & \mathbf{B}^- \\ \mathbf{0} & \mathbf{0} \end{bmatrix}}_{\tilde{\mathbf{A}}} \tilde{\mathbf{x}}[k] + \underbrace{\begin{bmatrix} \mathbf{B} \\ \mathbf{I} \end{bmatrix}}_{\tilde{\mathbf{B}}} \mathbf{u}[k] = \tilde{\mathbf{A}}\tilde{\mathbf{x}}[k] + \tilde{\mathbf{B}}\mathbf{u}[k], \\ \mathbf{y}[k] &= \underbrace{\begin{bmatrix} \mathbf{C} & \mathbf{0} \end{bmatrix}}_{\tilde{\mathbf{C}}} \tilde{\mathbf{x}}[k] = \tilde{\mathbf{C}}\tilde{\mathbf{x}}[k] \end{aligned}$$

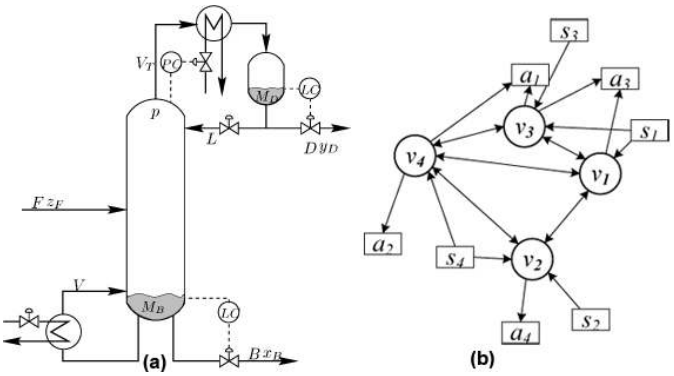


Figure 7: (a) Structure of the distillation column; (b) The network topology of the WCN corresponding to the sensor and actuator positions.

The above system has the same form as (1) and, therefore, we can use the aforementioned algorithms to obtain a stabilizing or optimal configurations of the WCN.

7. PROCESS CONTROL APPLICATION

The WCN has been deployed on a process-in-the-loop test-bed with a plant running in Simulink and the plant's sensors and actuators connected to analog interfaces (see Fig. 8(a)). We first describe the plant's model, then the closed-loop wireless control test-bed and finally demonstrate the WCN use for control of the plant.

7.1 Case Study Description

To illustrate the use of the WCN, we consider the distillation column control (Fig. 7(a)), a well-known process control problem described in [27]. Four input flows (in $[mols/s]$) are available for the column control: reflux (L), boilup (V), distillate (D) and bottom flow (B). The goal is to control four outputs: x_D - top composition, x_B - bottom composition, M_D - liquid levels in condenser, and M_B - liquid levels in the reboiler (in $[mol]$). Finally, the column has two disturbances, feed flow-rate F and feed composition z_F. The columns are described using the continuous-time Linear Time Invariant (LTI) model from [27], where the state-space contains 8 states.

7.2 WCN Experimental Platform

We have implemented the WCN scheme on FireFly embedded wireless nodes [28] and TI's MSP430F5438 Experimenter Boards, both equipped with IEEE 802.15.4 standard-compliant radio transceivers. FireFly is a low-cost, low-power platform based on Atmel ATmega1281 8-bit microcontroller, while the experimenters board uses a 16-bit MSP430 microcontroller. Both platforms can be used for TDMA-based communication with the RT-Link protocol [18], and support in-band synchronization provided as a part of the protocol.

The WCN procedure on each wireless node was implemented as a simple task executed on top of the nano-RK, a Real-Time Operating System (RTOS) [29]. The WCN task had a 140.64ms period, equal to the RT-Link frame size (RT-Link was configured to use 16 slots of size 8.79 ms). Since the WCN requires a TTA, nano-RK has been modified to enable scheduling of sensing and actuation at the start of the desired slots. This guarantees synchronized actions at all sensors and all actuators.

The column, modeled as a continuous-time LTI system along with disturbances and measurement noise was run in Simulink in real-time using Real-Time Windows Target [30]. The interface between the model and the real hardware were two National Instruments PCI-6229 boards which provided analog outputs that correspond to the Simulink model's outputs (see Fig. 8(a)). The output signals were saturated between -4V and 4V, due to NI boards limitations. Also, to provide inputs to the Simulink model, the boards sampled the analog input signals within range [-4V, 4V], at a 1 kHz rate. Finally, Simulink's input and output signals were monitored and controlled with 4 sensors and 4 actuators positioned according to the distillation column structure (Fig. 7(a)). In addition, 4 real wireless controller nodes ($v_1 - v_4$) were added, resulting in the topology shown in Fig. 7(b).

7.3 Results

From the communication and computation schedules, we obtained the discrete-time plant model using the discretization procedure from Section 6 (Eqs. (26),(27)), with sampling rate $T = 140.64\ ms$ (RT-Link frame size).

We first investigated the problem of providing MSS of the closed-loop system with uncorrelated random link failures and single node failures. Assigning each node to maintain a scalar state, using the procedures from Sections 4 and 5 we derived a stabilizing WCN configuration for the topology presented in Fig. 7(b) and the discretized LTI plant model. To solve the convex optimization problems we used the `CVX`, a package for specifying and solving convex programs [31].

We were able to obtain only WCN configurations that maintain stability if one of the nodes v_1-v_3 fails, meaning that the constraint from (23) for the node v_4 was violated (without v_4 the topology violates the conditions from [14], for existence of a stabilizing configuration). Fig. 9 shows obtained measurements where the disturbance inputs F, z_F were set to zero, while we provided periodical pulses to the input L. Although the output of the plant degrades when the node v_1 is turned off, the WCN maintains system stability. However, if the node v_4 is turned off, the system becomes unstable (shown in Fig. 10 - after the node is turned back

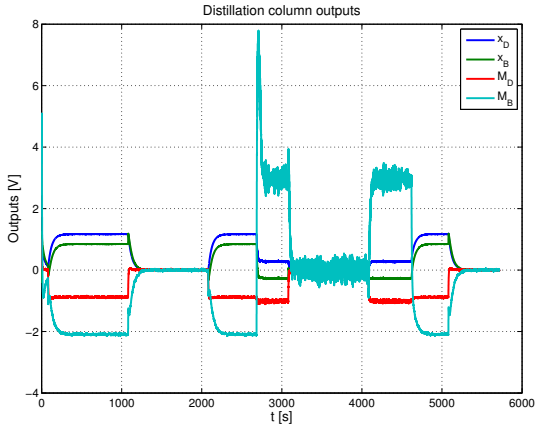
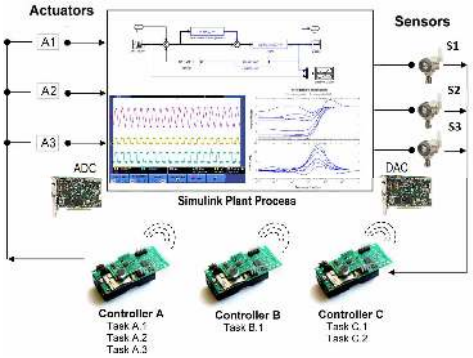


Figure 9: Plant outputs for a stabilizing WCN configuration. Node v_1 has been turned off at time $t = 1680\ s$ and turned back on at $t = 4560\ s$.



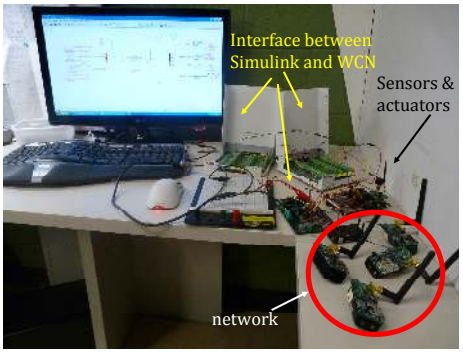


Figure 8: Process-in-the-loop simulation of the distillation column control; (a) The plant model is simulated in Simulink, while the WCN is implemented on FireFly nodes; (b) Experimental setup used for the WCN validation.

on, the system slowly, due to the output saturation, returns to stability). Finally, we showed that if a node was added, connected to actuator a_2, sensor s_4 and nodes v_2, v_4, we could maintain stability if one of the node fails.

We also considered optimal WCN design that minimizes effects of disturbance inputs F, z_F. Using Algorithm 1 we computed an optimal WCN configuration for energy to peak minimization. The obtained measurements for a setup with periodical F impulses are shown in Fig. 11. Fig. 11(b) and Fig. 11(a) present the plant outputs for the optimal and stable WCN configurations. As shown in Fig. 11(c), the norm of the output controlled with the optimal configuration is almost 5 times smaller than the norm with the stabilizing WCN.

8. CONCLUSION

We have extended the concept of the Wireless Control Network, where the network itself acts as a fully distributed controller. We have first addressed the WCN synthesis problem to guarantee *optimal* performance of the plant with respect to standard cost functions. Second, by including the observer style updates in the simple, linear iterative procedure, we have been able to significantly increase *robustness* of the closed-loop system to link failures. We have also proposed a method to extract a stabilizing configuration for the WCN that can deal with node failures. Finally, we have extended the synthesis procedure to deal with continuous-time plants, and demonstrated how the WCN can be used on an industrial application, using a process-in-the-loop setup with real hardware. In future, we aim to introduce complex control operations (e.g., Kalman filtering, model predictive control) and investigate heterogeneous nodes with varied computation/communication capabilities.

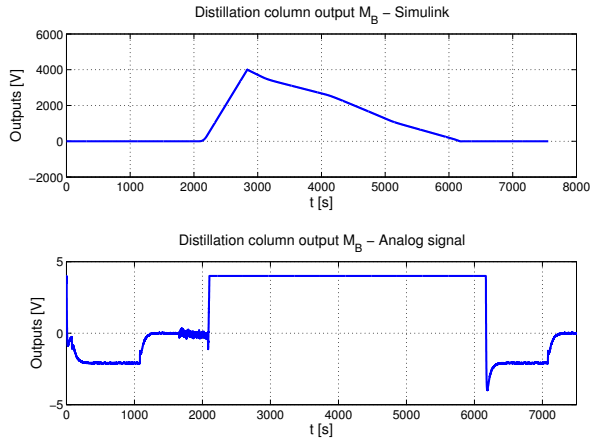


Figure 10: Distillation column output M_B. Node v_4 has been turned off at $t = 2140$ s and back on at $t = 2860$ s. Top - Simulink signal; bottom - analog signal, saturated at 4V.

9. REFERENCES

[1] HART Communication Foundation. Why WirelessHART? White Paper, Oct. 2007.

[2] S. Amidi and A. Chernoguzov. Wireless process control network architecture overview. White Paper, Mar. 2009.

[3] J. P. Hespanha, P. Naghshtabrizi, and Y. Xu. A survey of recent results in networked control systems. *Proceedings of the IEEE, Special Issue on Technology of Networked Control Systems*, 95(1):138 – 162, 2007.

[4] R. Alur, A.D'Innocenzo, K. H. Johansson, G. J. Pappas, and G. Weiss. Compositional modeling and analysis of multi-hop control networks. *IEEE Transactions on Automatic Control*, 56(10):2345–2357, Oct. 2011.

[5] W. Zhang, M.S. Branicky, and S.M. Phillips. Stability of networked control systems. *IEEE Control Systems Magazine*, 21(1):84–99, 2001.

[6] A. Saifullah, Y. Xu, C. Lu, and Y. Chen. Real-Time Scheduling for WirelessHART Networks. In *31st IEEE Real-Time Systems Symposium*, pages 150 –159, 2010.

[7] M. Pajic and R. Mangharam. Embedded virtual machines for robust wireless control and actuation. In *RTAS'10: 16th IEEE Real-Time and Embedded Technology and Applications Symposium*, pages 79–88, 2010.

[8] G. Fiore, V. Ercoli, A.J. Isaksson, K. Landernäs, and M. D. Di Benedetto. Multi-hop Multi-channel Scheduling for Wireless Control in WirelessHART Networks. In *IEEE Conference on Emerging Technology & Factory Automation*, pages 1 – 8, 2009.

[9] A. D'Innocenzo, G. Weiss, R. Alur, A.J. Isaksson, K.H. Johansson, and G.J. Pappas. Scalable scheduling

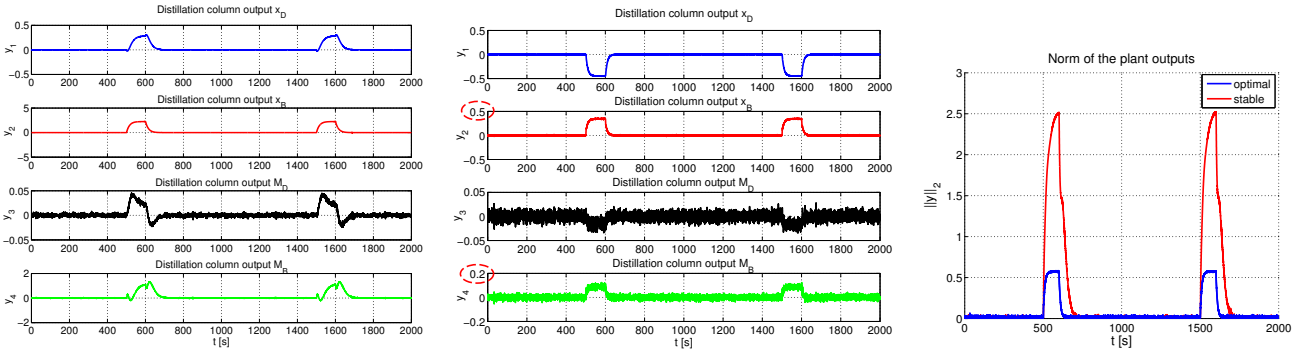


Figure 11: Distillation column outputs; (a) For a stable WCN configuration; (b) For an optimal WCN configuration (note the axes scales); (c) Comparison of the output vector norms for the stable and the optimal WCN configurations.

algorithms for wireless networked control systems. In *CASE'09: IEEE International Conference on Automation Science and Engineering*, pages 409–414, 2009.

[10] S. Graham, G. Baliga, and P.R. Kumar. Abstractions, architecture, mechanisms, and a middleware for networked control. *IEEE Transactions on Automatic Control*, 54(7):1490–1503, 2009.

[11] H. Kopetz and G. Bauer. The Time-Triggered Architecture. *Proceedings of the IEEE*, 91(1):112–126, 2003.

[12] M. Pajic, S. Sundaram, G. J. Pappas, and R. Mangharam. The Wireless Control Network: A New Approach for Control over Networks. *IEEE Transactions on Automatic Control*, 56(10):2305–2318, 2011.

[13] C. N. Hadjicostis and R. Touri. Feedback control utilizing packet dropping network links. In *Proceedings of the 41st IEEE Conference on Decision and Control*, pages 1205–1210, 2002.

[14] M. Pajic, S. Sundaram, G. J. Pappas, and R. Mangharam. Topological Conditions for Wireless Control Networks. In *Proceedings of the 50th IEEE Conference on Decision and Control*, pages 2353–2360, 2011.

[15] P. Seiler and R. Sengupta. Analysis of communication losses in vehicle control problems. In *Proceedings of the American Control Conference*, pages 1491–1496, 2001.

[16] N. Elia. Remote stabilization over fading channels. *Systems & Control Letters*, 54(3):237–249, 2005.

[17] Thomas Schmid, Prabal Dutta, and Mani B. Srivastava. High-resolution, low-power time synchronization an oxymoron no more. In *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, IPSN'10, pages 151–161, 2010.

[18] A. Rowe, R. Mangharam, and R. Rajkumar. Rt-link: A time-synchronized link protocol for energy-constrained multi-hop wireless networks. In *SECON'06: IEEE Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, pages 402 –411, 2006.

[19] ISA100.11a: Wireless systems for industrial automation: Process control and related applications. Standard, 2009.

[20] R. E. Skelton, T. Iwasaki, and K.M. Grigoriadis. *A unified algebraic approach to linear control design*. CRC Press, 1998.

[21] J. Han and R.E. Skelton. An LMI optimization approach for structured linear controllers. In *Proceedings of the 42nd IEEE Conference on Decision and Control*, pages 5143–5148, 2003.

[22] L. El Ghaoui, F. Oustry, and M. Ait Rami. A cone complementarity linearization algorithm for static output-feedback and related problems. *IEEE Transactions on Automatic Control*, 42(8):1171–1176, Aug. 1997.

[23] P.J. Antsaklis and A.N. Michel. *Linear Systems*. McGraw Hill, 1997.

[24] V. Gupta, A. F. Dana, J. Hespanha, R. M. Murray, and B. Hassibi. Data transmission over networks for estimation and control. *IEEE Transactions on Automatic Control*, 54(8):1807–1819, Aug. 2009.

[25] K. S.J. Pister and L. Doherty. Tsmp: Time synchronized mesh protocol. In *International Symposium on Distributed Sensor Networks (DSN)*, pages 391–398, 2008.

[26] S. Sundaram, M. Pajic, C.N. Hadjicostis, R. Mangharam, and G.J. Pappas. The Wireless Control Network: Monitoring for malicious behavior. In *Proceedings of the 49th IEEE Conference on Decision and Control*, pages 5979–5984, 2010.

[27] S. Skogestad and I. Postlethwaite. *Multivariable Feedback Control: Analysis and Design*. Wiley, 1996.

[28] R. Mangharam, A. Rowe, and R. Rajkumar. FireFly: A Cross-layer Platform for Real-time Embedded Wireless Networks. *Real-Time System Journal*, 37(3):183–231, 2007.

[29] The nano-RK Sensor Real-Time Operating System. `http://nanork.org`.

[30] Real-Time Windows Target - Run Simulink models on a PC in real time. http://www.mathworks.com/products/rtwt/.

[31] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, http://stanford.edu/ boyd/cvx, 2009.