4-2011

A Simple Distributed Method for Control over Wireless Networks

Miroslav Pajic
University of Pennsylvania, pajic@seas.upenn.edu

Shreyas Sundaram
University of Waterloo, ssundara@uwaterloo.ca

George Pappas
University of Pennsylvania, pappasg@seas.upenn.edu

Rahul Mangharam
University of Pennsylvania, rahulm@seas.upenn.edu

Suggested Citation:
Miroslav Pajic, Shreyas Sundaram, George J. Pappas and Rahul Mangharam, "A Simple Distributed Method for Control over Wireless Networks", CPS Week Workshop on Real-Time Wireless for Industrial Applications (RealWIN), Chicago, IL, 2011.

A Simple Distributed Method for Control over Wireless Networks

Abstract

We present a distributed scheme used for control over wireless networks. In our previous work, we introduced the concept of a Wireless Control Network (WCN), where the network itself, with no centralized node, acts as the controller. In this work, we show how the WCN can be modified to include observer style updates which substantially improves robustness of the closed-loop system to link failures. In addition, we analyze how the WCN simplifies extraction of the communication and computation schedules and enables system compositionality and scalability.

Disciplines

Controls and Control Theory | Electrical and Electronics | Systems and Communications

Comments

Suggested Citation:

Miroslav Pajic, Shreyas Sundaram, George J. Pappas and Rahul Mangharam, "A Simple Distributed Method for Control over Wireless Networks", CPS Week Workshop on Real-Time Wireless for Industrial Applications (RealWIN), Chicago, IL, 2011.

A Simple Distributed Method for Control over Wireless Networks

Miroslav Pajic*, Shreyas Sundaram†, George J. Pappas* and Rahul Mangharam*
*Dept. of Electrical & Systems Eng. †Dept. of Electrical & Computer Eng.
University of Pennsylvania University of Waterloo
{pajic, pappasg, rahulm}@seas.upenn.edu ssundara@uwaterloo.ca

***Abstract*—We present a distributed scheme used for control over wireless networks. In our previous work, we introduced the concept of a Wireless Control Network (WCN), where the network itself, with no centralized node, acts as the controller. In this work, we show how the WCN can be modified to include observer style updates which substantially improves robustness of the closed-loop system to link failures. In addition, we analyze how the WCN simplifies extraction of the communication and computation schedules and enables system compositionality and scalability.**

I. INTRODUCTION

Wireless Networked Control Systems (WNCSs) fundamentally differ from standard wired distributed systems as 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. The most important objective of WNCSs is to provide system stability, or in cases with unreliable communication, Mean Square Stability (MSS) of the closed-loop system. In addition, optimality (with respect to some appropriate cost function) is often desirable.

In traditional WNCSs, the network is used 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. The production of computationally more powerful wireless nodes has allowed the computation of the control algorithms to be migrated *into* the network, where a predefined node is assigned with the control algorithm execution. The use of wireless communication with such in-network computation requires runtime assignment of the control algorithm to any of the network nodes that satisfy the basic set of connectivity and computational requirements. The binding of the control algorithm to a specific controller (i.e., wireless node) makes the control infrastructure susceptible to failures of those (dedicated) nodes or any of the nodes used for data routing to and from the assigned controller. In these cases even a single-node failure might cause instability of the closed-loop system.

In [1], [2] we have recently introduced the Wireless Control Network (WCN), a fundamentally different concept where the network *itself* acts as the controller. In this work we present an improvement of the WCN, which bridges the gap between our previous work and existing networked control schemes, and substantially increases the system's robustness to link failures. In addition, we describe how the WCN can be used to decouple the extraction of communication and computation schedules which guarantee stability of the system.

II. WIRELESS CONTROL NETWORKS

In [1], [2] we have defined a WCN for the system presented in Fig. 1, where the plant is to be controlled using a multi-hop, fully synchronized wireless network with N nodes. We focus on discrete-time plants of the form:

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

with $\mathbf{A} \in \mathbb{R}^{n\times n}, \mathbf{B} \in \mathbb{R}^{n\times m}$ and $\mathbf{C} \in \mathbb{R}^{p\times n}$.[1] The output vector $\mathbf{y}[k]$ contains plants' output measurements provided by the sensors $s_1, \dots, s_p$, while the input vector $\mathbf{u}[k]$ corresponds to the signals applied to the plant by actuators $a_1, \dots, a_m$.

Unlike traditional networked control schemes where a particular node is designated as the controller (and all other nodes are used to route information between the node and the plant), with the WCN the network itself acts as a

[1]A similar model can be used for linear time-invariant, continuous-time plants. In this case a dynamical model of the closed-loop system can be obtained after the plant is discretized with a sampling period T, where the actuation delay is also taken into account [2]. For the purpose of clarity, we will use the WCN model with the discrete-time plant.

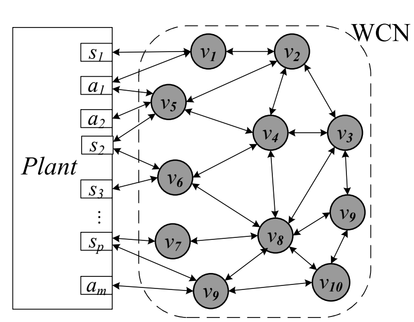
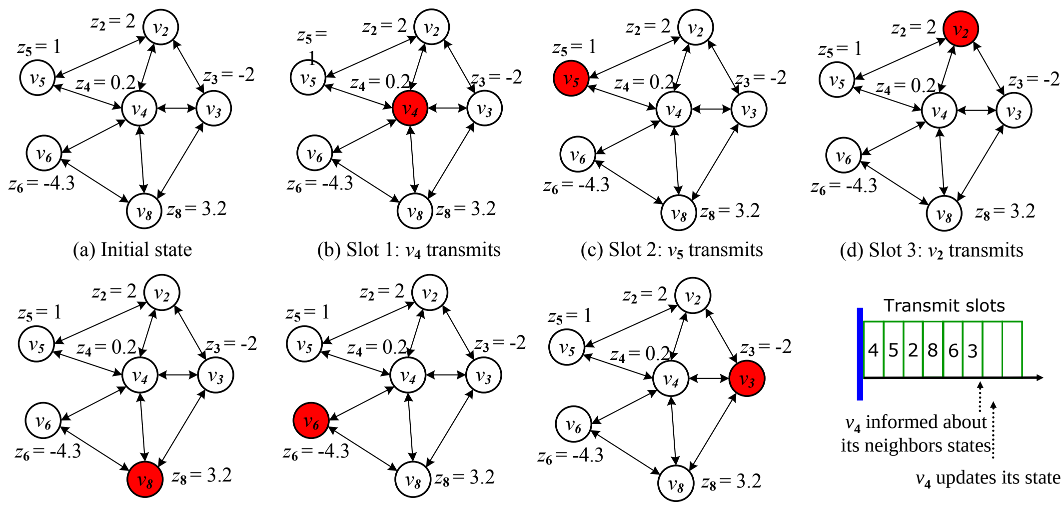


Figure 1. A multi-hop wireless control network used as a distributed controller.



(a) Initial state (b) Slot 1: v_4 transmits (c) Slot 2: v_5 transmits (d) Slot 3: v_2 transmits

(e) Slot 4: v_8 transmits (f) Slot 5: v_6 transmits (g) Slot 7: v_3 transmits (h) Communication schedule

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

controller, with no centralized node. This is achieved by having each node maintain a (possibly vector) state. All nodes execute a simple, linear iterative procedure, where in each time-step each node updates its value to be a linear combination of its previous value and the values of its neighbors. In addition, as described in [1], the update procedure includes a linear combination of the sensor measurements (i.e., plant outputs) from all sensors in the node's neighborhood.

For example, consider a network presented in Fig. 2, where at the beginning of a frame each node has an initial state value denoted with z_i (Fig. 2(a)). In the first time slot in a frame (Fig. 2(b)) node v_4 transmits its state, in the second node v_5 transmits the state, etc. Finally, in the 6^{th} slot node v_3 is the last node in the frame to transmits its state (Fig. 2(g)). This results in a communication schedule as depicted in Fig. 2(h). As can be noticed, 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 k, the update procedure can be expressed as:

$$z_i[k+1] = w_{ii}z_i[k] + \sum_{v_j \in \mathcal{N}_{v_i}} w_{ij}z_j[k] + \sum_{s_j \in \mathcal{N}_{v_i}} h_{ij}y_j[k], \quad (2)$$

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. Finally, each plant input $u_i[k]$ is computed as a linear combination of states from the nodes in the neighborhood of actuator a_i:

$$u_i[k] = \sum_{j \in \mathcal{N}_{a_i}} g_{ij}z_j[k]. \quad (3)$$

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 $\mathbf{z}[k]$, the network can be described as a dynamical compensator:

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

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

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

for all $k \in \mathbb{N}$. Since for all $i \in \{1, \dots, N\}$, $w_{ij} = 0$ 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. 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:

$$\hat{\mathbf{x}}[k+1] = \begin{bmatrix} \mathbf{x}[k+1] \\ \mathbf{z}[k+1] \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{A} & \mathbf{BG} \\ \mathbf{HC} & \mathbf{W} \end{bmatrix}}_{\hat{\mathbf{A}}} \begin{bmatrix} \mathbf{x}[k] \\ \mathbf{z}[k] \end{bmatrix} = \hat{\mathbf{A}}\hat{\mathbf{x}}[k]. \quad (4)$$

Several advantages of the proposed scheme are presented in [1], [2]. The WCN is computationally very inexpensive

Figure 3. A procedure used to extract a stabilizing configuration for the Wireless Control Network.

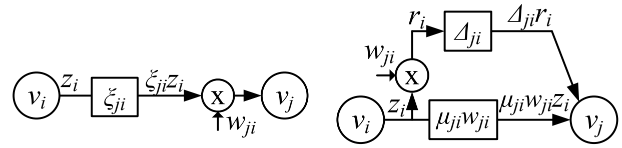


Figure 4. Communication over non-deterministic channel; (a) A link between nodes v_i and v_j; (b) Link transformation into a robust control form.

as each node only computes a linear combination of its value and values of its neighbors. This makes it suitable for even resource constrained, low-power wireless nodes (e.g., Tmote or FireFly [3]). In addition, the WCN can easily handle plants with multiple geographically distributed sensors and actuators, a case that is not easily handled by the "sensor $\rightarrow$ channel $\rightarrow$ controller/estimator $\rightarrow$ channel $\rightarrow$ actuator" setup that is commonly adopted in networked control design. The WCN does not rely on the existence of dedicated controllers, and inherently captures the case of nodes exchanging values with the plant at various points in the network. The WCN requires a simple transmission schedule where each node is active only once during a TDMA cycle. Furthermore, it allows the network operator to decouple the computation schedule from the communication schedule, as is described in Section IV.

A. *Finding a Stable Configuration*

To be able to use the WCN scheme it is essential to derive a procedure to determine an appropriate set of link weights which would result in a stable closed-loop system.[2] From Eq. (4), the closed-loop system is stable if the matrix $\hat{\mathbf{A}} = \hat{\mathbf{A}}(\mathbf{W}, \mathbf{H}, \mathbf{G})$ has all of its eigenvalues inside the unit circle. As the matrices $\mathbf{W}, \mathbf{H}, \mathbf{G}$ are structured, an additional set of constraints is imposed, which prevents us from directly applying standard design procedures based on Lyapunov and Linear Matrix Inequalities (LMI). However, an algorithm is provided in [1], [2]) to find a stabilizing configuration (a set of link weights which guarantees stability of the closed-loop system) in certain cases.

III. Improving robustness of the Wireless Control Network

In this section we consider robustness of the WCN to node and communication link failures.

A. *Robustness to link failures*

Since unreliability of the wireless communication links is one of the main drawbacks when wireless networks are used for control, a more 'realistic' system model was derived in [1], [2] that takes into account potential message drops. In addition, the procedure used to extract a stabilizing configuration can be adjusted to guarantee mean square stabilization despite unreliable communication links. The procedure uses information about network topology and a snapshot of current network conditions (i.e,. link qualities) to compute a stabilizing network configuration (Fig. 3). A derived stabilizing configuration can guarantee mean square stability of the system under packet drops in the network where all links are described as independent Bernoulli processes.

To improve WCN robustness to independent link failures we have modified the initial WCN scheme presented in Eq. (2), (3) to include observer style updates (as in [4]). In this case, the update procedure for each node v_j in the network is:

$$z_j[k+1] = w_{jj} z_j[k] + \sum_{i \in \mathcal{N}_{v_j}} (w_{ji} z_i[k] - q_{ji} z_j[k]), \quad (5)$$

where q_{ji} is a newly introduced set of link weights.[3]

Using the approach proposed in [5], each unreliable link $\xi_{ij} = (v_i, v_j)$ can be modeled as a memoryless, discrete, independent and identically distributed (IID) random process ξ, where IID implies that the random variables $\{\xi[k]\}_{k \geq 0}$ are IID. For each link, these random processes map each transmitted value t_{ji} into a received value $\xi[k] t_{ji}$ (see Fig. 4). In this work, we consider a model where each link (i.e., random process ξ_{ij}) is described as Bernoulli process with probability $p \leq 1$, meaning that the link (i.e., channel) will deliver the transmitted message with probability p.

Following the approach described in [5] each link described with a random process ξ_{ij} can be specified with a fixed gain and zero-mean random part: $\xi_{ij} = \mu_{ij} + \Delta_{ij}$ (the same approach was used in the initial WCN design). Therefore, the above update procedure becomes:

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

For each link $t = (v_i, v_j)$ we denote $r_t[k] = (w_{ji} z_i[k] - q_{ji} z_j[k])$. 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

[2]In this work, matrices $\mathbf{W}$, $\mathbf{H}$ and $\mathbf{G}$ that satisfy the topological constraints and guarantee stability of $\hat{\mathbf{A}}$ are referred to as a *stabilizing configuration*.

[3]A similar update is introduced for nodes that receive sensor values. This part has been omitted for simplicity.

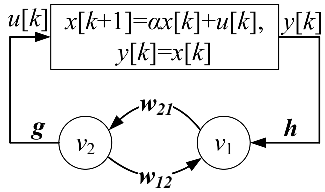


Figure 5. An example of WCN, a plant with a scalar state controlled by a WCN.

vector $\mathbf{r}[k]$ of length N_l (where N_l is the number of links), we obtain:

$$\mathbf{r}[k] = \mathbf{J}^{or}\begin{bmatrix}\mathbf{y}[k]\\ \mathbf{z}[k]\end{bmatrix} = \underbrace{\mathbf{J}^{or}\begin{bmatrix}\mathbf{C} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}_N\end{bmatrix}}_{\hat{\mathbf{J}}^{or}}\hat{\mathbf{x}}[k], \qquad (6)$$

where each row of the matrix $\mathbf{J}^{or} \in \mathbb{R}^{N_l\times(N+p)}$ contains either one or two nonzero elements, equal to a gain w_t, h_t, g_t or $-q_t$.[4]

This approach has enabled the use of similar method as for the 'basic' WCN to model the behavior of the closed-loop system. Specifically, the update equation for each node v_j is:

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

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

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

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

$$\mathbf{z}[k+1] = \mathbf{W}_\mu\mathbf{z}[k] + \mathbf{H}_\mu\mathbf{y}[k] + \mathbf{J}_v^{dst}\mathbf{\Delta}[k]\mathbf{r}[k],$$
$$\mathbf{u}[k] = \mathbf{G}_\mu\mathbf{z}[k] + \mathbf{J}_u^{dst}\mathbf{\Delta}[k]\mathbf{r}[k],$$

where all elements of matrices $\mathbf{W}_\mu, \mathbf{H}_\mu$ and $\mathbf{G}_\mu$ (except the diagonal entries of $\mathbf{W}_\mu$) are defined as in [1]. The diagonal entries of $\mathbf{W}_\mu$ are of the form $w_{jj} - \sum_{i\in\mathcal{N}_{v_j}}\mu_{ij}q_{ji}$. The binary (0, 1) matrices $\mathbf{J}_v^{dst}$ and $\mathbf{J}_u^{dst}$ are also defined as in [1], [2], where each row of the matrices selects elements of the vector $\mathbf{\Delta}[k]\mathbf{r}[k]$ that are added to the linear combinations calculated by the nodes and the actuators. Therefore, a procedure based on LMIs (similar to the previously described) can be used in this case to compute a stabilizing configuration for the WCN, which guarantees MSS of the closed-loop system.

[4]It is worth noting here that the difference from the 'basic' WCN case in [1] is that the latter design procedure contains only one nonzero element in each row of the matrix $\mathbf{J}^{or}$.

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

Figure 6. Maximal message drop probability which guarantees MSS for the system in Fig. 5 ($\alpha = 2$) without (WCN) and with observer style updates (oWCN)

	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)
$N=2$		$p_m = 20.40\%$	$p_m = 20.48\%$	$p_m = 20.64\%$

Figure 7. Maximal message drop probability which guarantees MSS for the system in Fig. 5 ($\alpha = 2$) with reliable link between node v_1 and actuator (observer style updates - oWCN)

1) Evaluation: To analyze robustness of the WCN with observer style updates we analyzed the performance of a WCN with $N \geq 2$ nodes that create a complete graph. The WCN is used for control of a single-state plant shown in Fig. 5 (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 taken to be a scaled version of the transmission of the node v_2 (i.e., $u[k] = gz_2[k]$, for some scalar g). Using the bisection method described in [2], we extracted the maximal probabilities of message drops (p_m) for which there exists a stabilizing configuration that guarantees 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 considered a network where the link between plant output and node v_1 is also unreliable. The results are presented in Fig. 6. In addition, we have investigated the case where the link between the plant's sensor and node v_1 is reliable (without any packet drops). The results are presented in Fig. 7. As can be noticed, the proposed scheme **significantly** improves system robustness to link failures. For example, the WCN with observer style updates can guarantee MSS for the system from Fig. 5 even when the probability of link failures is more than 20% (compared to 1.5% for the 'basic' WCN).

B. Robustness to Node Failures

The stability of the closed-loop system, described by Eq. (4), can be affected by node crash failures (nodes that stop working and drop out of the network). Currently, we have considered two approaches to deal with node failures. One obvious method to deal with up to k node failures is to precompute 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 (e.g., 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 [6] or the neighbors of failed nodes can 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 node failures would be to design the WCN in a way that even if some of its nodes fail the closed-loop system remains stable. For simplicity, consider a WCN that can deal with a single node failure. In this case let us denote with $\hat{\mathbf{A}}_i$ a matrix $\hat{\mathbf{A}}$ from Eq. (4) in the situation 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 $\mathbf{G}$. Thus,

$$\hat{\mathbf{A}}_i \triangleq \begin{bmatrix} \mathbf{A} & \mathbf{B}\mathbf{G}\mathbf{I}_N^i \\ \mathbf{I}_N^i\mathbf{H}\mathbf{C} & \mathbf{I}_N^i\mathbf{W}\mathbf{I}_N^i \end{bmatrix},\ i = 1, \ldots, N, \qquad (7)$$

where $\mathbf{I}_N^i$ 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 $\mathbf{X}$ (and, thus, a common Lyapunov function $V(\hat{\mathbf{x}}) = \hat{\mathbf{x}}^T\mathbf{X}\hat{\mathbf{x}}$) such that $\mathbf{X} - \hat{\mathbf{A}}^T\mathbf{X}\hat{\mathbf{A}} \succ \mathbf{0}$ and

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

Therefore, the procedure from [1] can be used to extract a stabilizing configuration that can deal with a single node failure, with additional N LMI constraints. However, in this case it is necessary to design the network in a way that guarantees that such stabilizing configuration exists. Initial results regarding these topological conditions have been presented in [7].

IV. Scheduling Communication and Computation for the WCN

The WCN scheme requires that each node transmits exactly once per frame. In addition, since the only requirement is that the communication schedule is collision-free, it is possible to schedule more than one node to transmit in a time slot, as long as as the nodes' transmissions do not interfere.

The communication schedule can be obtained from the network's interference graph using standard graph coloring techniques (as in [8]). If d_i is the maximal degree of the interference graph, the only constraint is that each frame contains at least d_i slots. It is worth noting here that the obtained communication schedule is **static** even in cases where some of the links significantly change their quality. The reason is that with the WCN there is no standard data routing to and from the controller, which would require the use of different routes when the network conditions change.

In the WCN each node v_i is assigned with a WCN task that can be described with a tuple (T, Φ^i, d^i, C^i) corresponding to the task's period, offset, relative deadline and the worst-case execution time. The period of the task T is the same for all nodes in the network and it is equal to the communication frame size. In the general case, the worst case execution time for the WCN tasks can differ from node to node. However, if all nodes in the network implement the same type of controllers, we can take $C^i = C$. The task offset Φ^i and deadline d^i can be derived from the communication schedule since the computation for each node can be scheduled only after the node and all of its neighbors had transmitted their states. Here, it is necessary to guarantee that the computation is performed before the node's next scheduled transmission (in the next frame). For example, in each frame, node v_j from Fig. 8(b) is scheduled to transmit last among all neighbors of node v_i. Thus, for the WCN task on node v_i:

$$\begin{aligned} \Phi^i &= T_{x_j} \cdot T_{slot} \\ d^i &= T + T_{x_i} \cdot T_{slot} \end{aligned}$$

where T_{x_i}, T_{x_j} are indexes of the transmit slots for nodes v_j, v_i and T_{slot} denotes the slot size.

This decoupling of the schedule extractions significantly simplifies schedulability analysis and enables compositional design of the WCN.

A. Composing new control loops

The most significant advantage of the WCN is that compositionality is inherent in the WCN since an introduction of a new control loop (e.g., plant) does not affect the performance of the existing control algorithms. For example, if the WCN is used to control P plants, it is possible to add a new plant to be controlled if the following conditions are met:

1. Each node can transmit all of its $P+1$ states in a single communication packet.
2. It is possible to schedule calculation of the $(P+1)^{st}$ linear combination without affecting the previous P calculations.

The WCN does not require data to be routed to and from the controller. Thus, the introduction of new loops does not involve any changes in the existing communication schedule. However, it is necessary to guarantee that all states can

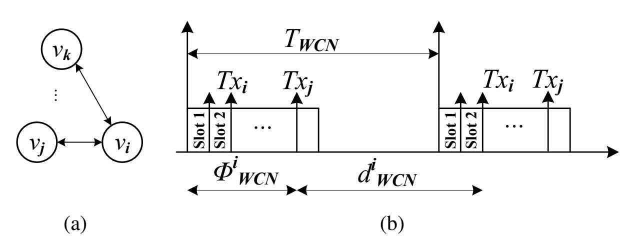


Figure 8. (a) Topological connectivity of node v_i; (b) An example of the scheduling for node v_i.

be transmitted in a single packet. Thus, for each node v_i we introduce a communication budget CB^i that is equal to the number of unused bytes in v_i's transmission packet. For example if low-bandwidth 802.15.4 protocol is used for communication for each node v_i, initially $CB^i = 128$.[5] For the case when node v_i maintains a scalar 16-bit state, adding a new loop would decrease its communication budget by 2 bytes.

If computation of an additional linear combination is to be assigned to each node in the network, instead of adding a new tasks we can consider a model where the execution time of the WCN task is increased by ΔC_{WCN}, where ΔC_{WCN} represents the worst case execution time that is needed to compute the additional linear combinations.[6] If each node executes only the WCN task, we can define a *computational budget* $CP^i = d^i - C^i$. In this case, computation of the new linear combination can be added to the WCN task if and only if $CP^i \geq \Delta C_{WCN}$.

The proposed scheduling method provides a sufficient condition for the schedulability analysis. However, it is worth noting that decoupling of the communication and computation scheduling might result in a non-optimal scheduling algorithm. Finding an optimal scheduling policy for the WCN will be an avenue of future work.

V. Industrial Application

To illustrate the use of the WCN we consider a well-known process control problem, the distillation column control described in [9] and shown in Fig. 9. Four inputs are available for the column control. These are flows: reflux (L), boilup (V), distillate (D) and bottom flow (B). The goal is to control 4 outputs: x_D - top composition, x_B - bottom composition, M_D - liquid levels in condenser, and M_B - liquid levels in the reboiler.

In our experiments, the distillation column inputs and outputs are monitored/controlled with 4 sensors and 4 actuators positioned according to the distillation column structure (Fig. 9). In addition, 4 nodes ($v_1 - v_4$) have been added, resulting in the network topology shown in Fig. 10. To demonstrate the performance of the WCN when used for the distillation column control we utilize the continuous-time Linear Time Invariant (LTI) model of the plant from [9]. The state-space model contains 8 states and the aforementioned 4 inputs and 4 outputs. Assigning each node to maintain a scalar state, using an extension of the procedure from [1], [2] we obtained a stabilizing configuration for the topology presented in Fig. 10 and a discretized LTI model of the distillation column. The stabilizing configuration is able to

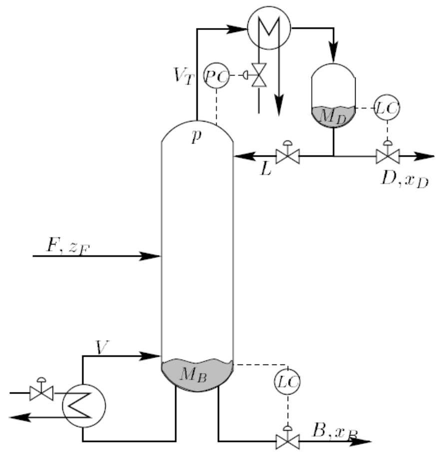


Figure 9. Structure of the distillation column. The column design and diagram are taken from [9].

guarantee mean square stability of the closed-loop system with uncorrelated random link failures and single node failures.

A. WCN Experimental Platform

The WCN scheme is implemented using FireFly embedded wireless nodes [3]. FireFly is a low-cost, low-power platform based on Atmel ATmega1281 8-bit microcontroller with 8KB of RAM and 128KB of ROM along with a Chipcon CC2420 IEEE 802.15.4 standard-compliant radio transceiver. FireFly nodes can be used for real-time TDMA-based communication with the RT-Link protocol [10]. RT-Link on FireFly nodes supports both tight global, hardware-based, out-of-band time synchronization and in-band synchronization provided as a part of the TDMA-based link protocol.

The linear iterative procedure on each wireless node is implemented as a simple task executed on top of the nano-RK RTOS [11]. On FireFlies, nano-RK operates with a 1ms OS tick. The WCN task was assigned a period equal to the RT-Link frame size. Since RT-Link was configured to use sixteen 5ms slots, the period of the WCN task is 80ms. In addition, since nano-RK and RT-Link provide support for Rx and Tx slot allocation at each node,

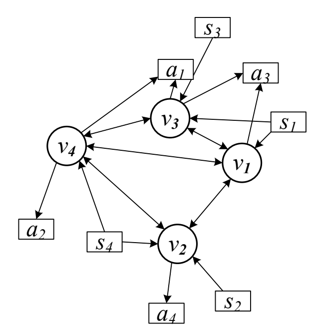


Figure 10. Structure of the WCN. The network topology corresponds to sensor and actuator positions in Fig. 9

[5] In the general case, node states do not have to be scalars as described in [2]. In addition, the nodes can have different state sizes, meaning that nodes might have different communication budgets.

[6] As computation of the linear combination does not require a task with high execution time, introducing a new task would be inefficient due to the overhead associated with each new task.

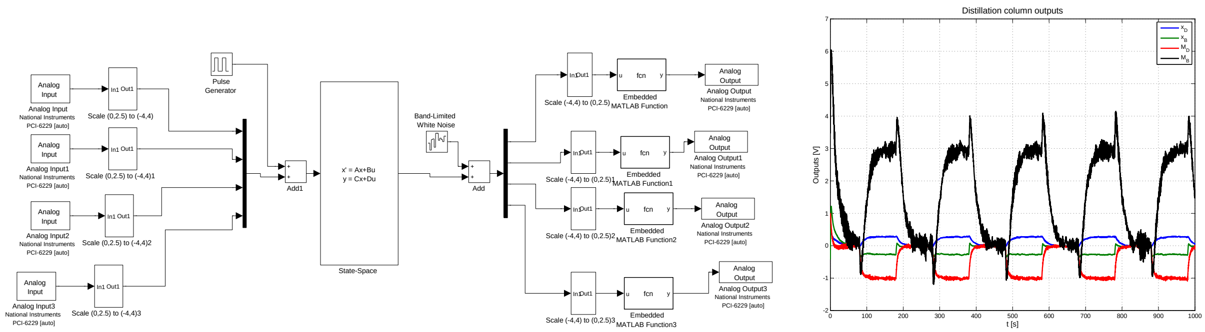


Figure 11. (a) Simulink design of the distillation column with data acquisition and analog to digital interfaces; (b) Outputs of the distillation column, x_D - blue, x_B - green, M_D - red, M_B - black line.

scheduling the computations (of the WCN procedure) and transmission slots was straightforward. Also, since the WCN requires Time-Triggered Architecture (TTA), nano-RK has been modified to enable scheduling of sensing and actuation at the beginning of the desired slots. This guarantees loosely synchronized sampling/actuation at all sensors and actuators, respectively.

The column, modeled as a continuous-time LTI system along with disturbances and measurement noise (see Fig. 11(a)), is run in Simulink in real-time using Real-Time Windows Target [12]. The interface between the Simulink model and the real hardware are two National Instruments PCI-6229 boards which provide analog outputs that correspond to the Simulink model's outputs (see Fig. 12). The output signals are saturated between -4V and 4V, due to NI boards limitations. The boards also sample the analog input signals within range [-4V, 4V], generated by the actuator nodes at a rate of 1 kHz to provide inputs to the Simulink model. Finally, Simulink's input and output signals are processed by the WCN with the topology from Fig. 10. Using the aforementioned setup, we have been able to demonstrate that the WCN is able to maintain stability of the distillation column, while maintaining only a scalar state at each of the four wireless nodes. Fig. 11(b) presents outputs of the distillation column. As can be noticed, although the output M_D initially overshoots the upper bound (i.e., 4V) the WCN is able to eventually stabilize the plant.

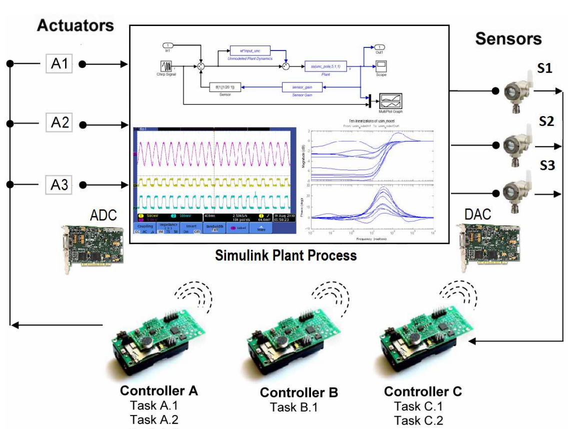


Figure 12. Process-in-the-loop simulation of the distillation column control; The plant model is simulated in Simulink, while the Wireless Control Network is implemented on FireFly nodes.

VI. Limitations and Future Work

While the proposed scheme has several benefits in comparison to traditional control schemes, there are also some drawbacks which will be addressed through future research. We discuss some of these below.

• In our analysis of WCN robustness to link failures we have assumed independent link failures in the network (both in time and in space). On the other hand, in some applications links could have correlated losses, and might not be memoryless. In these cases the procedure presented in this work could not be used to guarantee MSS of the system. However, other works on networked control (such as [4]) have studied methods of dealing with arbitrary models of link failures, and it will be of interest to extend our design algorithms to such cases.

• Our scheme to handle node failures (described in Section III) can be applied only if the network topology satisfies the requirements for which there exists a stabilizing configuration that maintains stability if up to k nodes fail. However, it is desirable to develop a scheme for each node to follow to adapt its link weights in situations when the topology changes. Currently, the procedure used in this work requires solving an LMI optimization problem at each iteration, which makes it unsuitable for implementation on resource constrained nodes. In addition, we do not have a characterization of the number of iterations required for our algorithms to converge (although the number of variables in the optimization algorithms scales well with the number of nodes and links - quadratically and linearly, respectively). We were able to obtain a promising result for positive systems[7] that cover a large class of dynamical systems [13]. In this case, a stabilization configuration can be obtained by solving a set of linear matrix inequalities (which can be solved in polynomial time).

[7] A system is said to be positive if the state $\mathbf{x}[k]$ and output $\mathbf{y}[k]$ from Eq. (1) are confined to non-negative orthant for all $k \geq 0$ if the initial state $\mathbf{x}[0]$ is non-negative.

• This paper assumes that the topology of the WCN is specified *a priori*, and presents a numerical algorithm to design the linear weights for each node. The dual approach of finding appropriate topologies that will be capable of stabilizing a given system is described in [7]. In addition, it would be desirable to find procedure for network synthesis such that we are able to extract a WCN configuration that mimics the behavior of a predefined controller as closely as possible.

VII. Conclusion

We have extended the concept of the Wireless Control Network, where the network itself acts as a fully-distributed controller. To increase the robustness of the WCN with respect to link failures we have modified the initial WCN procedure. 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. In addition, we have proposed a method to extract a stabilizing configuration for the WCN that can deal with node failures. We have also demonstrated how the WCN enables compositional system design and simplifies extraction of compatible communication and computation schedules. Finally, on the distillation column case study we have shown how the WCN can be used for control in real-world process control applications.

Acknowledgment

The authors thank to Paul McLaughlin and Alexander Chernoguzov from Honeywell Process Solutions for fruitful discussions during the preparation of this paper. This research has been partially supported by the NSF-CNS 0931239 and NSF-MRI Grant 0923518. It has also been supported by a grant from NSERC.

References

[1] M. Pajic, S. Sundaram, J. Le Ny, G. J. Pappas, and R. Mangharam, "The Wireless Control Network: Synthesis and Robustness," in *Proceedings of the 49th IEEE Conference on Decision and Control*, 2010, pp. 7576–7581.

[2] M. Pajic, S. Sundaram, R. Mangharam, and G. J. Pappas, "The Wireless Control Network," Technical Report, 2010.

[3] R. Mangharam, A. Rowe, and R. Rajkumar, "FireFly: A Cross-layer Platform for Real-time Embedded Wireless Networks," *Real-Time System Journal*, 2007.

[4] 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*, vol. 54, no. 8, pp. 1807–1819, Aug. 2009.

[5] N. Elia, "Remote stabilization over fading channels," *Systems & Control Letters*, vol. 54, no. 3, pp. 237–249, 2005.

[6] 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*, 2010, pp. 5979 –5984.

[7] 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*, 2011, submitted.

[8] J. Kleinberg and E. Tardos, *Algorithm Design*. Addison-Wesley, 2005.

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

[10] A. Rowe, R. Mangharam, and R. Rajkumar, "RT-Link: A Time-Synchronized Link Protocol for Energy-Constrained Multi-hop Wireless Networks," *IEEE SECON*, 2006.

[11] "The nano-RK Sensor Real-Time Operating System," http://nanork.org.

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

[13] L. Farina and S. Rinaldi, *Positive linear systems: theory and applications*. John-Willey & Sons, Inc., 2000.