2017 15th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)

Implementation and Characterization of a Multi-hop 6TiSCH Network for Experimental Feedback Control of an Inverted Pendulum

Craig B. Schindler*, Thomas Watteyne†, Xavier Vilajosana‡, Kristofer S. J. Pister*
*Department of Electrical Engineering and Computer Sciences
University of California, Berkeley, Berkeley, California, USA
Email: {craig.schindler,ksjp}@berkeley.edu
†EVA team, Inria, Paris, France
Email: thomas.watteyne@inria.fr
‡Wireless Networks Research Group, Universitat Oberta de Catalunya, Barcelona, Spain
Email: xvilajosana@uoc.edu

***Abstract*—6TiSCH is a technology being standardized at the IETF which brings determinism to low-power wireless communication. In a 6TiSCH network, all communication is orchestrated by a communication schedule. This paper explores the applicability of this new technology to control systems. In particular, we apply it to the inverted pendulum, a canonical control system in which a cart moves along a track to keep a pendulum – which naturally falls over – upright. This paper presents the first characterization and implementation of a closed-loop wireless feedback control network using completely standards-compliant IEEE802.15.4 TSCH technology. First, we implement a control loop in OpenWSN and experimentally evaluate the performance of the network by varying the radio duty cycle, number of hops, and introducing controlled external interference. We show that 100% reliability can be achieved while maintaining latencies well below the critical delay of the system. Second, we use the network on an inverted pendulum system and show that angular deviations from the upright position do not exceed 3 degrees, even in a multi-hop setup. Finally, we discuss the results in detail, and advocate for a co-design of the controller and the networking system.**

I. INTRODUCTION

Standards such as WirelessHART [1] and IEEE802.15.4 TSCH [2] have introduced wire-like reliability to low-power wireless multi-hop mesh networks. Commercial products now exist which offer over 99.999% end-to-end reliability and over a decade of lifetime on a pair of AA batteries [3]. The result is that these standards have been adopted massively by industrial applications; tens of thousands of these networks[1] operate today to monitor refinery tank farms, monitor bearing temperature in sugar processing plants, monitor water usage in pharmaceutical production plants, etc.

These standards all rely on the principle of Time Synchronized Channel Hopping – nodes in a network synchronize, time is organized in timeslots, and a communication schedule orchestrates all communication. Because TSCH uses time synchronization, radios are on only a fraction of the time (<1% is typical), leading to extremely long battery lifetime. Because TSCH uses channel hopping, the network is resilient to external interference and multi-path fading, leading to extremely high reliability. Both lifetime and reliability are exploited fully in the industrial process monitoring applications listed above. The IETF 6TiSCH working group [4][2] was created in 2013 to standardize a protocol stack which captures the industrial performance of these networks, while allowing them to seamlessly integrate into the Internet.

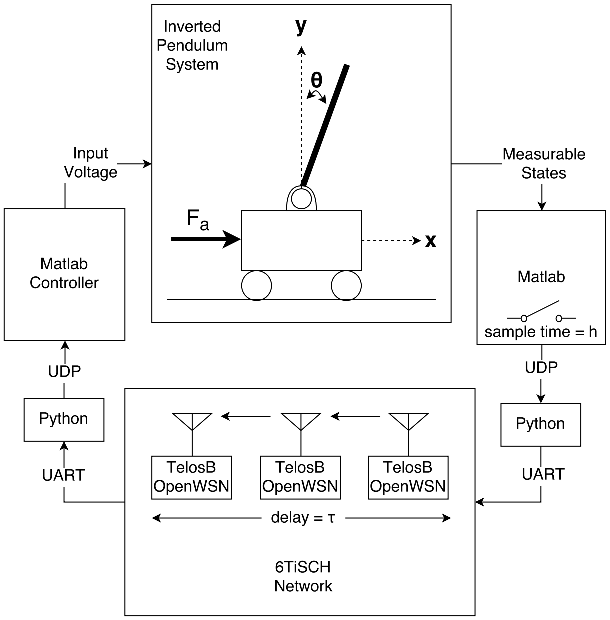

Fig. 1. Block diagram of the wirelessly controlled inverted pendulum system with a 2-hop 6TiSCH network.

This paper is a very first step in the *industrial process*

---

[1] One vendor alone – Emerson Process Management – announced over 27,800 networks deployed.

[2] IPv6 over the TSCH mode of IEEE 802.15.4e Working Group, https://tools.ietf.org/wg/6tisch/charters

978-3-901882-90-6/17

monitoring to *industrial process control* transition. With it, we want to answer the question: *Can 6TiSCH be used as a networking technology for running a control loop?* In this paper, we use the canonical "inverted pendulum" control system, in which a cart moves along a track to self-balance a stick which otherwise falls over. Typically, an inverted pendulum is sampled at 1 kHz, with essentially no delay or packet loss. We replace the wires of the system with a multi-hop 6TiSCH network (see Fig. 1), thereby necessarily introducing delay, jitter, and limiting the system to sample at most 10-100Hz. While previous work has been done demonstrating wireless feedback control of the inverted pendulum (Section II-A), this is the first time it has been demonstrated using entirely standards-based IEEE802.15.4 TSCH and 6TiSCH technology.

The contributions of this paper are:

1) We create a 6TiSCH network using OpenWSN, capable of satisfying the inverted pendulum latency requirements, and we experimentally characterize the radio duty cycle, end-to-end reliability, and end-to-end latency while changing the communication schedule, increasing the number of hops, and adding external interference.
2) We show successful experimental stabilization of an inverted pendulum using this 6TiSCH network, with a maximum angle deviation with respect to the upright position of less than 3 degrees, even in a multi-hop setup.

II. CONTROLLING AN INVERTED PENDULUM

Stabilizing and controlling an inverted pendulum is a classic and well understood engineering problem. The top block of Fig. 1 illustrates what an inverted pendulum system is. The goal of the controller is to stabilize the pendulum in the upright position by modulating the force on the cart to which the pendulum is attached and free to rotate in the x-y plane. The upright (inverted) position of the pendulum is an unstable equilibrium point, and therefore a feedback control law is required for stabilization. Many more complex systems can be modeled as an inverted pendulum, including human balance and rockets [5], [6].

A. Previous Work on Wireless Inverted Pendulum Control

Hernandez et al. have demonstrated stabilization of an inverted pendulum over IEEE802.15.4 using TelosB motes running TinyOS [7]; however, there were a number of issues preventing the implementation from being standards-compliant. Even more importantly, their implementation was created before the IEEE802.15.4e amendment was created, which standardized channel hopping. Ploplys et al. have demonstrated stabilization of an inverted pendulum over IEEE802.11, a technology which is unsuitable for low-power applications [8]. Hörjel has demonstrated stabilization of an inverted pendulum over Bluetooth Classic, which is also a technology unsuitable for any type of low-power application [9].

B. Inverted Pendulum Model

The linearized coupled equations of motion describing the cart position $x(t)$ and the pendulum angle $\theta(t)$ are given in (1), with M the mass of the cart, m the mass of the pendulum, l the distance between the pendulum's pivot point on the cart and the pendulum's center of mass, and $F_a(t)$ the force exerted on the cart in response to the input voltage applied to the cart's motor. Fig. 7 shows the experimental setup we are using: a single rigid pendulum setup by Quanser[3]. A more detailed derivation of (1) as well as numerical details of the Quanser system can be found in [10].

$$\begin{aligned} (M+m)\ddot{x} + ml\ddot{\theta} &= F_a \\ ml\ddot{x} + \frac{4}{3}ml^2\ddot{\theta} - mgl\theta &= 0 \end{aligned} \tag{1}$$

The two equations in (1) can be represented in state space form using (2), where the state vector $q(t) \in \mathbb{R}^{4\times1}$ contains the system's four states: the cart position $x(t)$, the cart velocity $\dot{x}(t)$, the inverted pendulum angle $\theta(t)$, and the inverted pendulum angular velocity $\dot{\theta}(t)$. The matrix $A \in \mathbb{R}^{4\times4}$ expresses the relationship between the system states and the system state derivatives. The scalar $u(t)$ represents the input voltage to the cart's motor. The matrix $B \in \mathbb{R}^{4\times1}$ expresses the relationship between the input voltage to the cart's motor and the system state derivatives.

$$\begin{gathered} \dot{q} = Aq + Bu \\ A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -6.8123 & -1.4957 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 15.4731 & 25.6566 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1.5226 \\ 0 \\ -3.4583 \end{bmatrix} \end{gathered} \tag{2}$$

C. Full-State Feedback Control

Stability of the system in (2) is determined by the eigenvalues of A. The system is stable if all of the eigenvalues of A are in the left half-plane, and the system is unstable otherwise. A full state feedback control law is shown in (3).

$$u = -kq \tag{3}$$

When the linear system in (2) is controlled using the full-state feedback law in (3), the resulting system stability is determined by the eigenvalues of $(A-Bk)$, where $k \in \mathbb{R}^{1\times4}$ is a constant vector. The system is stable if all of the eigenvalues of $(A-Bk)$ are in the left half-plane, and the system is unstable otherwise [6]. The entries of k are generally calculated using the Linear Quadratic Regulator (LQR) algorithm, which is the optimal solution of k [11].

D. Critical Delay

All wireless feedback control systems have some finite time delay τ and sample time h. The time delay τ is defined as the amount of time between when the system state is sampled and when the controller receives the sample. The sample time h is defined as the amount of time between two subsequent samples of the system state. It is known that both large time delays and large sample times can introduce oscillations into nominally stable systems and potentially cause instability [6], [12]. In general, stability of linear time-invariant (LTI) systems with

---

[3] http://www.quanser.com/

constant time delay can be analyzed using delay differential equation theory, or a rational approximation of the time-delay transfer function [6]. Similarly, stability of LTI systems with constant sampling time can be analyzed using sampling theory [12]. Matlab can also be used to numerically investigate these analytic techniques.

Previous work on the inverted pendulum system [5], [13], [14] discusses a *critical sensor to actuator delay* τ_c, which if sustained in the feedback loop too long causes instability. The closed-form expression of τ_c given by Stépán and Kollár is $\sqrt{(2l)/(3g)}$, with l the distance between the pendulum's pivot point on the cart and the pendulum's center of mass, and g the acceleration due to gravity. This analytic value of τ_c sits well with one's intuition that a long stick is easier to balance than a short stick.

While theoretical work has been done on networked control systems with variable time delay and sample time, it is still an area open for new research [15], [16].

The distance between the pendulum's pivot point and center of mass used in our experiments is $l = 0.33$ m, and therefore $\sqrt{(2l)/(3g)} = 150$ ms. As shown in Section IV, the implemented 6TiSCH networks used in our experiments have nominal time delay and sample time well below this value.

Qualitatively, if the time delay of the feedback network is too large for too long, then the controller is unable to respond faster than the internal dynamics of the inverted pendulum, and the system becomes unstable. Likewise, if the sample time of the feedback network is too large for too long, the cart may move too much between samples, and the system may become unstable. Understanding the dynamics and latency requirements of a system is crucial for its dependability.

III. 6TiSCH Low-Power Wireless Network

Time Synchronized Channel Hopping (TSCH) [17] is at the core of virtually all industrial low-power wireless standards, including WirelessHART [1], ISA100.11a [18] and IEEE802.15.4-2015 TSCH [2].

In a TSCH network, time is divided into timeslots (each typically 10-15 ms long). Timeslots are grouped into a slotframe (typically 10-1000 timeslots long), which continuously repeats over time. Each timeslot is long enough for a node to transmit a data packet to its neighbor, and for that neighbor to send back an acknowledgment indicating successful reception. A communication schedule orchestrates all communication in the network. It indicates to each node what to do in each timeslot: transmit, listen or sleep. For each slot in which the node communicates, the schedule also indicates on which frequency.

IEEE802.15.4-2015 [2] is the latest standard which adopts TSCH as one of its core medium access techniques. The IETF 6TiSCH working group [4] currently standardizes mechanisms to build this schedule and match it to the application's needs. The "minimal" approach [19] is the simplest option: all nodes have the same schedule, which consists of a number of active timeslots, followed by a number of timeslots in which the nodes' radios stay off (see Fig. 2)[4]. All active timeslots are equivalent, and marked as `TX`, `RX`. This results in the behavior:

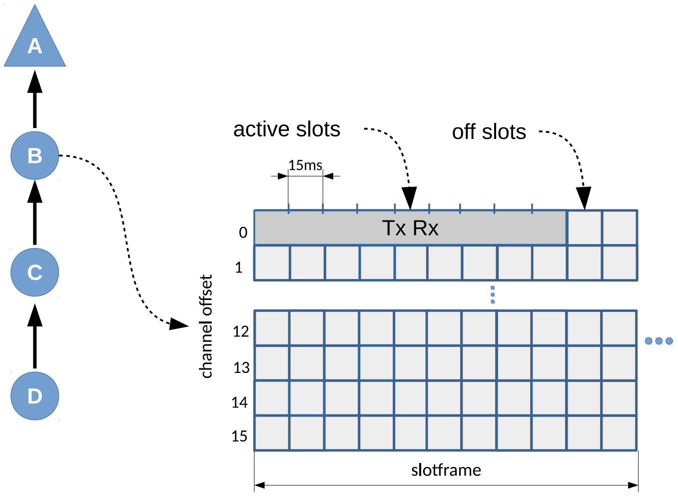

Fig. 2. The 6TiSCH "minimal" communication schedule, shown here with an 11-slot slotframe and 9 active slots.

- If a node doesn't have data frames to transmit, it listens.
- If a node transmits data to a neighbor and does not receive an acknowledgment, a back-off mechanism is used to resolve collisions.

All link-layer frames (both unicast and broadcast) are transmitted in these timeslots, which are all equivalent. The RPL routing protocol is used to organize the network topology in a multi-hop routing structure, as shown on the left of Fig. 2. When D has a packet to send to A, the packet advances by one hop at each active timeslot, except when link-layer retransmissions are needed.

This slotted structure offers basic link-layer connectivity, in a slotted-Aloha fashion, while introducing channel hopping. On top of this, 6TiSCH is standardizing mechanisms for neighbor nodes to establish dedicated slots to one another in a distributed fashion [20], [21]. And while it is clear that using dedicated slots yields (far) better performance than shared "minimal" slots, we stick with the latter in this paper. The idea is that if the system performs well with this simple configuration, it can only perform better when more advanced mechanisms are used. We use OpenWSN [22] to create our network, the reference open-source 6TiSCH implementation.

The OpenWSN project was started at the University of California, Berkeley, and implements a fully standards-based protocol stack, including IEEE802.15.4-2015 TSCH, 6TiSCH, 6LoWPAN, RPL and CoAP [23], [24]. It has been ported to 11 hardware platforms, ranging from low-end 16-bit microcontrollers to powerful ARM Cortex-M architectures. OpenWSN provides an ecosystem of tools to interface low-power wireless mesh networks with the Internet. The open-source nature allows us to instrument the code to measure reliability, latency

---

[4] Although all timeslots are scheduled on channel offset 0, this translates to a different frequency at every timeslot – using a pseudo-random function – resulting in channel hopping [4].

and jitter, while varying the communication schedule, the number of hops and introducing external interference.

IV. CHARACTERIZING NETWORK PERFORMANCE

Before using a 6TiSCH network with the real inverted pendulum system, we want to benchmark the network's performance and verify that its nominal end-to-end latency is lower than the critical delay of the pendulum derived in Section II-D. We characterize the network performance of an OpenWSN 6TiSCH network as described in Section III, using the "minimal" schedule depicted in Fig. 2 – outside of the pendulum context. We are particularly interested in the following performance criteria:

- (end-to-end) *reliability*: the portion of the packets sent by the source node which reach their final destination.
- (end-to-end) *latency*: the amount of time between the moment the source node injects a data packet into the network, until the moment it reaches its final destination.
- *jitter*: the spread of the latency.

We use the schedule depicted in Fig. 2, with a slotframe of 11 timeslots. We vary the following network configurations, and measure the impact of those changes on the performance criteria listed above:

- *schedule*: we vary the number of active slots in the slotframe, either 2, 5, 8 or 11.
- *hop count*: we increase the number of "hops" the packet needs to travel to go from the source to the destination, from 1 to 4.
- *interference*: we add up to 3 "jammer" nodes into the system, each one programmed to introduce continuous external interference on a particular frequency.

Experimental results are summarized in Table I. Details about this table are given in the following sections.

A. Experimental Setup

Fig. 3 depicts the experimental setup. TelosB motes are programmed with the OpenWSN firmware. They are colocated, and the OpenWSN `topology.c` utility is used to force a linear multi-hop topology by filtering link-layer frames based on their source address. A computer is connected to both the source and the destination nodes through a USB cable. For every experiment, the computer sends 1000 packets through the network, and for each packet measures whether it was received, as well as the end-to-end latency.

B. About Reliability

When targeting industrial applications, a wireless network *must* offer wire-like reliability. This is because a low-power wireless system typically replaces a wired network. For the team of engineers in charge of the industrial process, it is inconceivable to start loosing packets "just" because the system is now wireless.

Most industrial standards, for example HART7, have set user availability expectations at 99.9% [25]. Table I shows that the network meets this requirement in all cases.

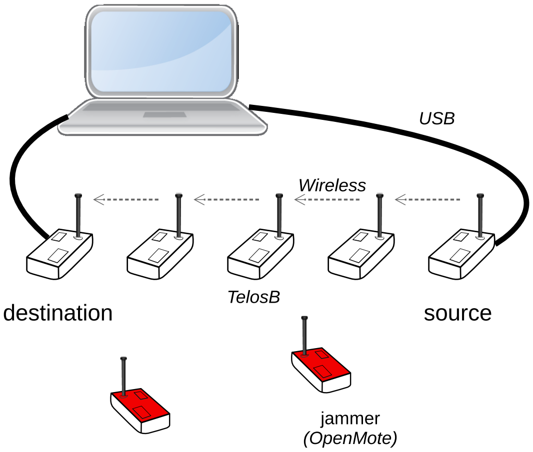

Fig. 3. Experimental setup used to characterize network performance.

C. Network Performance Results

A total of 10 experiments are carried out, each involving sending 1000 packets through the network. High-level statistics about each experiment are presented in Table I. The "hops" column indicates the number of wireless hops, i.e. the number of nodes minus one. The "duty cycle" column indicates the radio duty cycle of the nodes (the average portion of time the radio is on), specifying in each case how many active slots this corresponds to. We use the energy consumption model from [26] to translate the number of active slots to radio duty cycle. The reported duty cycle of these experiments is calculated with respect to an average of one packet being sent through the network per slotframe. The "blocked channels" column lists the frequency channels the different jammer nodes operate on. IEEE channel notation is used, i.e. channel 11 corresponds to 2.405 GHz. The "reliability" column indicates the percentage of packets that have made it through the system and details the number of transmitted and received packets. The "latency" column lists the minimum, median and maximum latency, computed on all the packets that were successfully received.

The high-level latency statistics from Table I are complemented by the histogram of the different latencies, shown as "violin plots" in Figs. 4, 5, and 6. Bars on these plots show the minimum, median and maximum latency values, i.e. the same values as in Table I[5].

We start with a favorable "baseline" setup in which all 11 slots in the slotframe are active, with only a single hop, and no jammers present. The left-most violin plot in Figs. 4, 5, and 6 are the same, and represent this baseline. To plot Fig. 4, we re-program the motes so their schedule contains less active slots. Having less active slots means there are more off slots, which increases the latency as data is buffered longer. To plot Fig. 5, we add nodes to obtain a linear network with different numbers of hops. Latency increases linearly with hop count,

---

[5] Occasional OpenWSN error codes resulted in timing and recording a very small number of latency outliers.

hops	**duty cycle**	**blocked channels**	**reliability (numRx/numTx)**	**latency (min/med/max)**
Baseline				
1	25% (11/11 slots)		100.0% (1000/1000)	33 ms / 50 ms / 106 ms
Impact of Schedule				
1	18% (8/11 slots)	[]	100.0% (1000/1000)	31 ms / 56 ms / 142 ms
1	12% (5/11 slots)	[]	100.0% (1000/1000)	31 ms / 116 ms / 285 ms
1	6% (2/11 slots)	[]	100.0% (1000/1000)	78 ms / 130 ms / 347 ms
Impact of Hop Count				
2	27% (11/11 slots)	[]	99.9% (999/1000)	50 ms / 65 ms / 317 ms
3	27% (11/11 slots)	[]	100.0% (1000/1000)	65 ms / 82 ms / 261 ms
4	27% (11/11 slots)	[]	99.9% (999/1000)	80 ms / 95 ms / 354 ms
Impact of Interference				
1	25% (11/11 slots)	[11]	99.9% (999/1000)	39 ms / 47 ms / 86 ms
1	25% (11/11 slots)	[11,16]	99.9% (999/1000)	35 ms / 47 ms / 237 ms
1	25% (11/11 slots)	[11,16,21]	100.0% (1000/1000)	33 ms / 47 ms / 207 ms

TABLE I
KEY PERFORMANCE STATISTICS FOR THE 6TiSCH NETWORK.

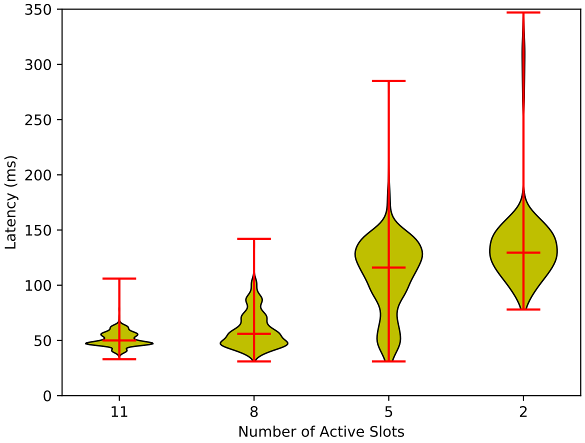

Fig. 4. Impact of the schedule (the number of active slots) on end-to-end latency.

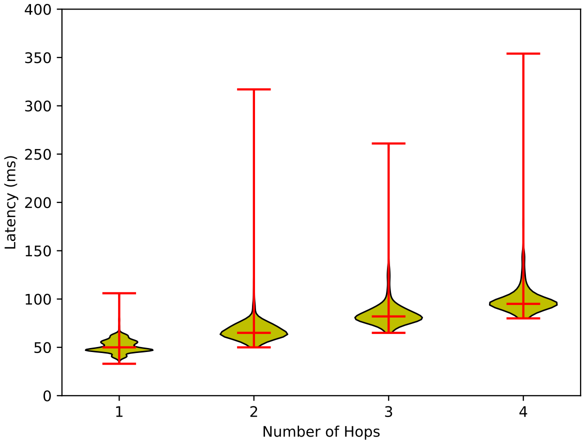

Fig. 5. Impact of hop count on end-to-end latency.

with each extra hop adding approximately 15 ms of latency (the duration of one timeslot when using TelosB motes). To plot Fig. 6, we place up to 3 jammer nodes in the direct vicinity of the communicating nodes. A jammer node is a node (here an OpenMote [27]) which runs the `oos_jammer` OpenWSN project. Once loaded, the node continuously sends frames of 127 bytes in length on a single channel, 246 packets per second, at +7 dBm. The frame contains random bytes. This causes the jammer operating frequency to be almost continuously "blocked". Fig. 6 shows that the presence of jammers (even blocking 3 out of 16 channels) has a very limited effect of the end-to-end latency, confirming the effectiveness of channel hopping to combat external interference.

It should be noted that external interference as well as the occasional transmission of heartbeat and maintenance packets between nodes as per standards-based requirements [19] may momentarily increase end-to-end latency of the network.

These results show that a 6TiSCH network offers performance in terms of reliability and latency better than what is needed for the inverted pendulum system under consideration.

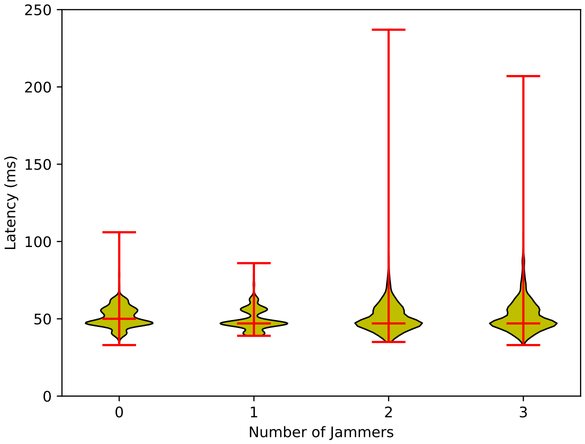

Fig. 6. Impact of external interference on end-to-end latency.

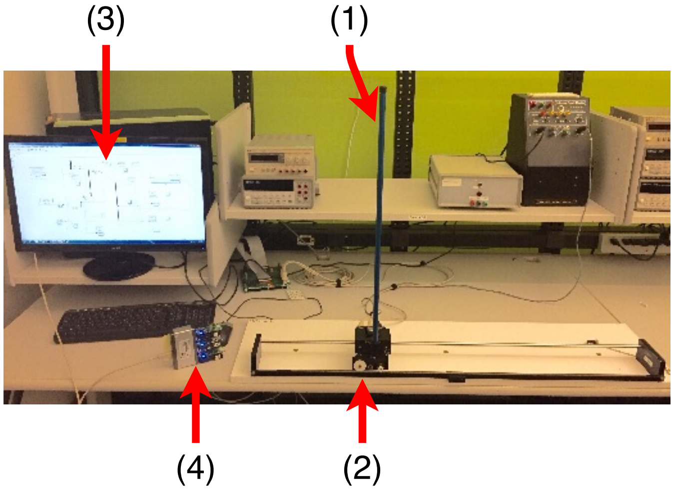

Fig. 7. A picture of the experimental setup including: (1) the inverted pendulum, (2) the cart, (3) the computer running the Matlab-based controller, and (4) the wireless motes.

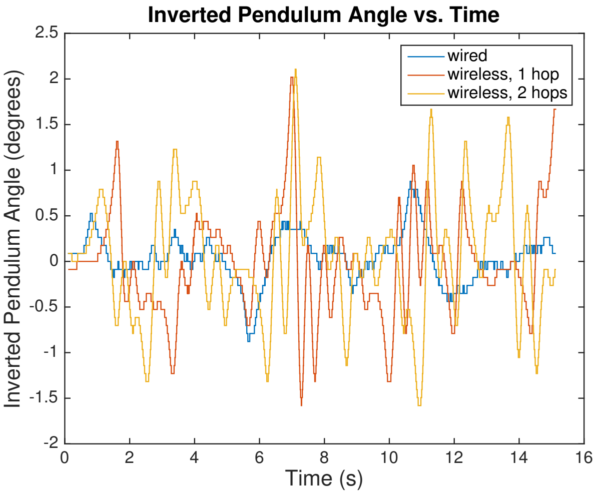

Fig. 8. Experimental data of inverted pendulum angle vs. time. Data for each of the three different setups is sampled at 1 kHz using the pendulum's built in rotary encoder. The resolution of the pendulum's rotary encoder is 0.09 degrees.

It hence confirms the applicability of 6TiSCH for this system.

V. (WIRELESS) CONTROL OF AN INVERTED PENDULUM

In a 6TiSCH network, the time delay τ is a random variable which follows distributions like those depicted in Figs. 4, 5, and 6. Our wireless implementation matches the sample time h to the expected value of the network's time delay τ so that a new packet can be sent across the network as soon as the previous packet is received by the controller. The pros and cons of schemes designed to reduce h such as pipelining and dropping stale data are discussed in Section VI.

A. *Experimental Setup*

A block diagram of the full experimental system can be seen in Fig. 1, and a picture of the actual setup in Fig. 7. The Quanser system includes the inverted pendulum, the cart (with two built-in rotary encoders that sample the cart position x and the pendulum angle θ both at 1kHz), and a DAC and electronic amplifier for powering the cart.

The following subsections report and compare the results from three different setups: wired feedback control, 1-hop wireless feedback control using 6TiSCH, and 2-hop wireless feedback control using 6TiSCH. The real-time controller is implemented in Matlab. Cart velocity $\dot{x}$ and pendulum angular velocity $\dot{\theta}$ are estimated by numerically differentiating x and θ, respectively, and smoothing using a 50 ms moving average filter. The k vector in the full-state feedback control law in (3) was calculated using Matlab's `lqr` function, with $Q = I_4$ and $R = 0.1$. The same k was used for all three experiments which allows us to compare relative performance.

A plot of inverted pendulum angle vs. time can be seen in Fig. 8. To compare the relative performance of the baseline (wired) feedback control and the 6TiSCH wireless feedback control, we collect experimental data of the inverted pendulum angle away from the upright position (Table II) and histograms of the inverted pendulum angle (Fig. 9). The upright position is defined as 0 degrees. Each experiment is 60 seconds long.

In the wireless setup, the TelosB motes in Fig. 1 run the OpenWSN firmware using a slot schedule with 11/11 active slots. The Matlab controller gathers the system's four state variables (see Section II-B), each of which are 8-byte double precision values. Matlab then injects this 32-bytes of data into the source mote's serial port. The source mote forwards the data to the destination mote through the 6TiSCH network. Once the destination mote receives the 32-bytes of data, it publishes it on its serial port, effectively injecting it (back) into Matlab. Matlab reconstructs the state vector and calculates the voltage to be applied to the cart's motor using (3). The DAC and electronic amplifier input this voltage into the cart's motor.

B. *Baseline: Wired Control*

The wired feedback control does not use the 6TiSCH network. The input voltage is calculated in Matlab without data being sent over the wireless network. The inverted pendulum system running the wired feedback control is stable, and of the three setups has the least amount of deviation from 0 degrees, as can be seen in Fig. 9.

C. *Single-Hop Wireless Control*

The 1-hop experiment consists of one source mote and one destination mote. The source mote sends the 32 bytes directly to the destination mote. The expected transmission latency across the 1-hop 6TiSCH network is approximately 45 ms as three time slots are needed in total: one for loading a packet over serial into the source mote, one for transmitting the packet from the source mote to the destination mote, and one for the destination mote to publish the packet over its serial port. A sampling block is used in the Matlab controller to match the sample time with the total expected time delay across the network. An extra 10 ms is added to the expected 6TiSCH

Feedback Type	Mean (degrees)	Std. Dev. (degrees)	Min (degrees)	Max (degrees)
wired	0.02	0.28	-0.88	0.88
wireless, 1-hop	-0.03	0.54	-1.67	2.02
wireless, 2-hops	0.05	0.74	-2.46	2.11

TABLE II

INVERTED PENDULUM ANGLE STATISTICS FOR WIRED FEEDBACK, 1-HOP WIRELESS FEEDBACK, AND 2-HOP WIRELESS FEEDBACK.

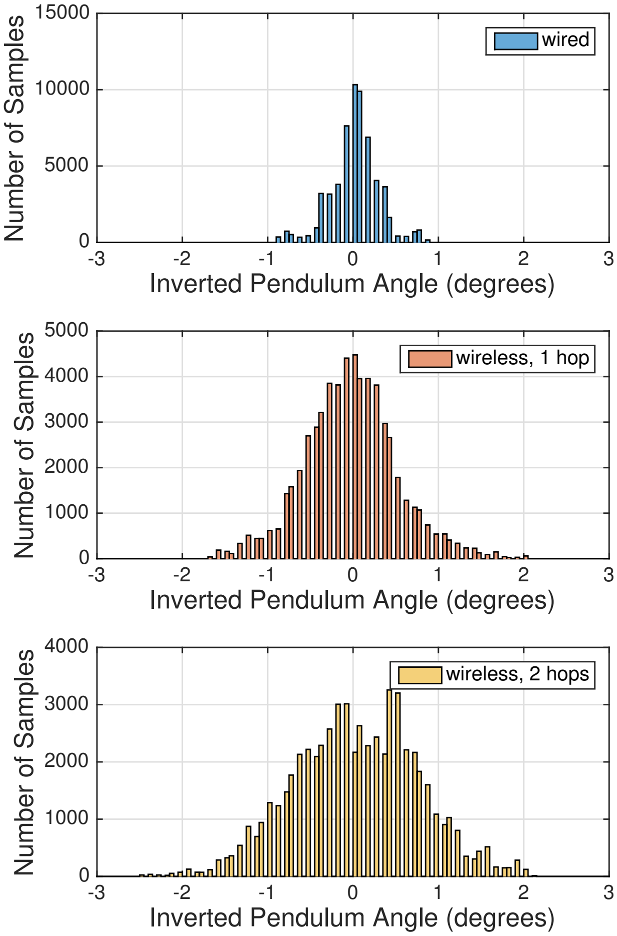

Fig. 9. Histograms of inverted pendulum angle from experimental data. Data for each of the three different setups is sampled at 1 kHz for 60 seconds using the pendulum's built-in rotary encoder. The resolution of the pendulum's rotary encoder is 0.09 degrees.

network time delay in order to buffer against additional unaccounted latency in the system, and therefore the sampling block broadcasts a new packet every 55 ms. The total expected time delay and sample time of the 1-hop network is therefore 55 ms.

The 1-hop wireless feedback control also proves to be very stable, with performance similar to the wired setup. The inverted pendulum angle ranged from -1.67 degrees to 2.02 degrees, only slightly more than the range of the wired feedback control (Fig. 9). Similarly, the standard deviation of the inverted pendulum angle using 1-hop wireless feedback control is slightly larger than when using wired feedback control (Table II).

D. Multi-Hop Wireless Control

The 2-hop experiment consists of one source mote, one relay mote, and one destination mote. The source mote sends the 32 bytes to the relay mote, and the relay mote then sends the 32 bytes to the destination mote. The expected transmission latency across the 2-hop 6TiSCH network is 60 ms because four time slots are needed in total: one for loading a packet over serial into the source mote, one for transmitting the packet from the source mote to the relay mote, one for transmitting the packet from the relay mote to the destination mote, and one for the destination mote to publish the packet over its serial port. A sampling block is used in the Matlab controller to match the sample time with the total expected time delay across the network. An extra 10 ms is added to the expected 6TiSCH network time delay in order to buffer against additional unaccounted latency in the system, and therefore the sampling block broadcasts a new packet every 70 ms. The total expected time delay and sample time of the 2-hop network is therefore 70 ms.

The 2-hop wireless feedback control also proves to be very stable, with performance similar to the wired and 1-hop wireless setups. The inverted pendulum angle ranged from -2.46 degrees to 2.11 degrees, only slightly more than the range of the wired and 1-hop wireless feedback control (Fig. 9). Similarly, the standard deviation of the inverted pendulum angle using 2-hop wireless feedback control is only slightly larger than when using wired or 1-hop wireless feedback control (Table II).

VI. DISCUSSION

This paper demonstrates successful closed-loop feedback control over a multi-hop 6TiSCH network, suggesting the applicability of 6TiSCH as a networking technology for running wireless control loops. This section discusses these results, their implications/pitfalls, and presents future work.

The implementation presented matches the sample time to the expected time delay of the network, so that a new packet enters the network as soon as the previous packet exits. A *pipelining* scheme in a multi-hop topology can further decrease the sample time by having multiple samples travel through the network at once. That is, as soon as the source mote transmits a packet to its parent mote, it indicates (through its serial port) that it is ready to receive a new state. This would require link-layer ACK/NACK (negative acknowledgment) to ensure end-to-end reliability in the presence of lossy links. While this scheme does not lower the time delay across the network, it lowers the sample time.

This paper's implementation is designed for ultra-high reliability, i.e. a mote that does not successfully transmit a packet re-transmits it. In some cases it might be better to intentionally drop an unsuccessfully transmitted packet so that transmission of a new (fresher) state sample can happen. The benefits of transmitting all samples successfully (whether stale or not) must be weighed against the benefit of having the controller have quick access to the most recent state sample. The number of packet retries can be tuned for the particular application and goals of the feedback control network, to *trade-off reliability for freshness*.

All three experimental setups were successfully able to recover from an external disturbance created by aggressively tapping the inverted pendulum during operation. While in this paper we characterize the general performance of the inverted pendulum system using wired and wireless feedback control, the tail of any wireless latency distribution is in theory infinite; system-level fail-safe mechanisms need to be investigated to understand the implication this has.

We believe the performance of our system can be pushed further by co-designing the controller and the network. That is, designing a new class of controller which takes into account the relatively low throughput of the 6TiSCH network, and the latency/jitter associated with the communication. This would extend previous work on controller design capable of accurate state estimation and prediction [28], [29], [30], [31], and we believe it would allow network delays above the critical delay of the system.

ACKNOWLEDGMENTS

This work is partially supported by the Berkeley Sensor & Actuator Center, the European Commission through the H2020 F-Interop and ARMOUR projects, Inria through the REALMS associate team, the Spanish Ministry of Economy and the FEDER regional development fund through the SINERGIA project (TEC2015-71303-R).

REFERENCES

[1] H. C. Foundation, "WirelessHART Specification 75: TDMA Data-Link Layer (HCF SPEC-75)," 2008.
[2] IEEE, "802.15.4-2015: IEEE Approved Draft Standard for Low-Rate Wireless Personal Area Networks (WPANs)," 2015.
[3] T. Watteyne, J. Weiss, L. Doherty, and J. Simon, "Industrial IEEE802.15.4e Networks: Performance and Trade-offs," in *IEEE International Conference on Communications (IEEE ICC), Internet of Things Symposium*. London, UK: IEEE, 8-12 June 2015.
[4] D. Dujovne, T. Watteyne, X. Vilajosana, and P. Thubert, "6TiSCH: Deterministic IP-enabled Industrial Internet (of Things)," *IEEE Communications Magazine*, vol. 52, no. 12, pp. 36–41, 2014.
[5] J. Milton, J. L. Cabrera, T. Ohira, S. Tajima, Y. Tonosaki, C. W. Eurich, and S. A. Campbell, "The Time-delayed Inverted Pendulum: Implications for Human Balance Control," *Chaos*, vol. 19, 2009.
[6] K. J. Åström and R. M. Murray, *Feedback Systems*. Princeton University Press, 2012.
[7] A. Hernández, J. Faria, J. Araújo, P. Park, H. Sandberg, and K. H. Johansson, "Inverted Pendulum Control over an IEEE 802.15.4 Wireless Sensor and Actuator Network," in *International Conference on Embedded Wireless Systems and Networks (EWSN)*, Bonn, Germany, 23-25 February 2011.
[8] N. J. Ploplys, P. A. Kawka, and A. G. Alleyne, "Closed-Loop Control over Wireless Networks," *IEEE Control Systems Magazine*, vol. 24, 2004.
[9] A. Hörjel, "Bluetooth in control," Master's thesis, Dept. of Automatic Control, Lund Institute of Technology, Lund, Sweden, 2001.
[10] *EE128, Lab 6a: Pole Placement for the Inverted Pendulum*, University of California, Berkeley, Spring 2015.
[11] *EE128, Lab 6c: LQR Controller Design for Inverted Pendulum*, University of California, Berkeley, Spring 2015.
[12] N. S. Nise, *Control Systems Engineering: Fourth Edition*. John Wiley & Sons, Inc., 2004.
[13] G. Stépán and L. Kollár, "Balancing with Reflex Delay," *Mathematical and Computer Modelling*, vol. 31, 2000.
[14] M. Landry, S. A. Campbell, K. Morris, and C. O. Aguilar, "Dynamics of an Inverted Pendulum with Delayed Feedback Control," *SIAM Journal on Applied Dynamical Systems*, vol. 4, 2006.
[15] P. Naghshtabrizi and J. P. Hespanha, "Stability of Networked Control Systems with Variable Sampling and Delay," in *Allerton Conference on Communication, Control, and Computing*, Urbana-Champaign, IL, USA, 24 July 2006.
[16] W. Zhang, M. S. Branicky, and S. M. Phillips, "Stability of Networked Control Systems," *IEEE Control Systems*, vol. 21, no. 1, pp. 84–99, 2001.
[17] K. Pister and L. Doherty, "TSMP: Time Synchronized Mesh Protocol," *IASTED Distributed Sensor Networks*, pp. 391–398, 2008.
[18] ISA, "ISA-100.11a-2009 Wireless Systems for Industrial Automation: Process Control and Related Applications," 2009.
[19] X. Vilajosana and K. Pister, *Minimal 6TiSCH Configuration*, Internet Engineering Task Force Std. draft-ietf-6tisch-minimal-16 [work-in-progress], 28 June 2016.
[20] Q. Wang and X. Vilajosana, "6top Protocol (6P)," Internet Engineering Task Force, Tech. Rep. draft-ietf-6tisch-6top-protocol-02 [work in progress], 25 July 2016.
[21] D. Dujovne, L. A. Grieco, M. R. Palattella, and N. Accettura, "6TiSCH 6top Scheduling Function Zero (SF0)," Internet Engineering Task Force, Tech. Rep. draft-ietf-6tisch-6top-sf0-01 [work in progress], 8 July 2016.
[22] T. Watteyne, X. Vilajosana, B. Kerkez, F. Chraim, K. Weekly, Q. Wang, S. Glaser, and K. Pister, "OpenWSN: a Standards-based Low-power Wireless Development Environment," *Transactions on Emerging Telecommunications Technologies (ETT)*, vol. 23, 2012.
[23] M. R. Palattella, N. Accettura, X. Vilajosana, T. Watteyne, L. A. Grieco, G. Boggia, and M. Dohler, "Standardized Protocol Stack for the Internet of (Important) Things," *IEEE Communications Surveys Tutorials*, vol. 15, no. 3, pp. 1389–1406, 2013.
[24] X. Vilajosana, P. Tuset-Peiro, F. Vazquez-Gallego, J. Alonso-Zarate, and L. Alonso, "Standardized Low-Power Wireless Communication Technologies for Distributed Sensing Applications," *Sensors*, vol. 14, no. 2, 2014.
[25] K. Pister, P. Thubert, S. Dwars, and T. Phinney, *Industrial Routing Requirements in Low-Power and Lossy Networks*, Internet Engineering Task Force Std. RFC5673, October 2009.
[26] X. Vilajosana, Q. Wang, F. Chraim, T. Watteyne, T. Chang, and K. Pister, "A Realistic Energy Consumption Model for TSCH Networks," *IEEE Sensors*, vol. 19, 2014.
[27] X. Vilajosana, P. Tuset, T. Watteyne, and K. Pister, "OpenMote: Open-Source Prototyping Platform for the Industrial IoT," in *EAI International Conference on Ad Hoc Networks (AdHocNets)*, San Remo, Italy, 1-2 September 2015.
[28] J. P. Hespanha and A. R. Mesquita, "Networked Control Systems: Estimation and Control over Lossy Networks," *Encyclopedia of Systems and Control*, pp. 842–849, 2015.
[29] D. Antunes, J. Hespanha, and C. Silvestre, "Stability of Networked Control Systems with Asynchronous Renewal Links: An Impulsive Systems Approach," *Automatica*, vol. 49, no. 2, pp. 402–413, 2013.
[30] G.-P. Liu, Y. Xia, J. Chen, D. Rees, and W. Hu, "Networked Predictive Control of Systems with Random Network Delays in both Forward and Feedback Channels," *IEEE Transactions on Industrial Electronics*, vol. 54, no. 3, pp. 1282–1297, 2007.
[31] B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, M. I. Jordan, and S. S. Sastry, "Kalman Filtering with Intermittent Observations," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1453–1464, 2004.