Wireless Event-Triggered Controller for a 3D Tower Crane Lab Process

Faisal Altaf, José Araújo, Aitor Hernandez, Henrik Sandberg and Karl Henrik Johansson

***Abstract*— This paper studies the design and real-time implementation of an event-triggered controller for a nonlinear 3D tower crane where the communication between the controller and the actuators is performed over a low-power wireless network. A flexible Event-Generation Circuit (EGC) is proposed in order to implement event-driven controllers for Networked Control Systems. Furthermore, a detailed experimental analysis on the performance of the event-triggered controller and the influence of packet losses on the transmitted actuation messages are presented. The results show that the event-triggered controllers in networked control systems are able to maintain the same level of performance as compared to periodic controllers, while increasing the sensors/actuators lifetime by reducing network bandwidth utilization.**

I. INTRODUCTION

Increasingly, control systems are operated over large-scale, networked infrastructures. The use of wireless communication technology provides major advantages in terms of increased flexibility, and reduced installation and maintenance costs. Following this trend, several vendors are introducing devices that communicate over low-power wireless sensor networks (WSNs) for industrial automation and process control. While WSNs have been widely analyzed and deployed to extract information from the physical world [1], actuation over wireless networks is still in its infancy. When performing control over wireless networks, the control engineers face the challenge of designing algorithms that can overcome the effects of delay, packet losses, limited bandwidth and mainly, limited battery lifetime of the sensors and actuators. In particular, most efforts of the network control systems community have been conducted under the assumption of periodic sampling and actuation [2], which, in general, may require data rates not practical in a wireless system. To address these issues, two new control paradigms have recently been proposed to efficiently use the communication resources in an aperiodic fashion while enforcing prespecified control performance: *event-triggered control* [3], [4], [5], [6] monitors the state of the plant to select the time instants at which the control input needs to be updated; while *self-triggered control* [7], [8], [9] represents a model-based emulation of event-triggered control. Both self-triggered and event-triggered control strategies are applicable to a large set of systems (including nonlinear), provide guarantees on the control performance (in the form of rates of convergence) and can accommodate bounded delays.

To the best of our knowledge, the only existing implementation of event-triggered control appeared in [10], where the authors performed control of a thermofluid process over a wired Ethernet channel. We present in this paper the first design of an Event-Generation Circuit (EGC) which can implement event-triggered controllers over wireless networks. This device is microcontroller-based, low-power and highly flexible, that can be used for remote event detection, but also as an event detector and controller in an event-triggered, self-triggered, or periodic manner. Due to its low cost, the EGC can be easily replicated and be integrated for the control of other physical systems such as Heating, Ventilation and Air Conditioning (HVAC) systems, autonomous vehicles and other industrial processes.

In previous studies, the analysis on the influence of the unreliable wireless network in an event-triggered controller has been performed in simulations for very simple plants [11], [12]. In this work, we evaluate the EGC when performing control of a real 3D tower crane lab process over the low-power IEEE 802.15.4 wireless network, where the packet delivery is affected by varying delays and packet losses. The wireless nature of the communication medium renders energy and network bandwidth considerations much more critical, and poses a set of new problems with respect to wired control. Moreover, we describe the required extensions of the event-triggered formulation in [5] and controller design for tracking step reference changes in the 3D tower crane.

The present work represents a major step towards the design of resource-aware implementations of control laws over Wireless Sensor and Actuator Networks (WSANs). We demonstrate the applicability of event-triggered control of a physical system over a WSAN and its advantages with respect to the traditional periodic paradigm, both in terms of energy savings and bandwidth reduction.

The experimental results presented in this paper show that for the considered tower crane system, the event-triggered strategy reduces the number of control update transmissions by 61% with respect to a conventional periodic strategy. Furthermore, we demonstrate that the designed event-triggered controller is robust to high packet losses and delays.

The rest of the paper is organized as follows. We revisit the methodology for performing event-triggered control and provide an extension of those results for the case of step reference tracking in Section II. Section III introduces the mathematical model of the 3D tower crane lab process and the control design for the case of event-triggered and periodic cases. The experimental setup and the design of the Event-Generation Circuit is described in Section IV and Section V presents the results on the implementation of event-triggered controllers over wireless. Finally, Section VI summarizes the main ideas and proposes possible extensions of the work.

The authors are with the ACCESS Linnaeus Center, Electrical Engineering, Royal Institute of Technology, Stockholm, Sweden. E-mails: {faltaf,araujo,aitorhh,hsan,kallej}@ee.kth.se. The work of the authors was supported by the EU project FeedNetBack, the Swedish Research Council, the Swedish Strategic Research Foundation, and the Swedish Governmental Agency for Innovation Systems.

II. Event-Triggered Control for Step Reference Tracking

This section provides an overview of the event-triggered control formulation proposed in [5] and the necessary extensions in order to perform event-triggered control for asymptotically tracking step references in dynamical systems.

In [5] the regulation problem has been solved under event-triggered sampling for linear systems:

$$\dot{x}(t) = Ax(t) + Bu(t), \quad y(t) = Cx(t) \tag{1}$$

Note that $x(t) \in \mathbb{R}^n$, $u(t) \in \mathbb{R}^m$, $y(t) \in \mathbb{R}^p$. Now, suppose the asymptotically stabilizing controller $u(t) = Kx(t)$ has already been designed in the continuous domain for system (1). The control signal $u(t)$ is computed at the last sampling time t_k as $u(t) = Kx(t_k)$, where K is an $m \times n$ controller gain matrix. Moreover, the state $x(t_k)$ is measured and sent to the controller at time instant t_k. Then, for the time $t \in [t_k, t_{k+1})$ between two measurements, the sensor measurement error $e(t) \in \mathbb{R}^n$ can be introduced as $e(t) = x(t_k) - x(t)$. Note that $e(t_k) = 0$ at every sampling time. In the presence of this measurement error the controller will be applying the control signal given by:

$$u(t) = K\left(x(t) + e(t)\right), \tag{2}$$

where $u(t)$ is held constant between sampling times $t \in [t_k, t_{k+1})$. Now, using (2) in the system (1) we get the following closed-loop system:

$$\dot{x}(t) = (A + BK)x(t) + BKe(t). \tag{3}$$

According to [5], using the Lyapunov input-to-state stability theorem (ISS), the control system (3) is guaranteed to be ISS with respect to the measurement error $e(t)$ if:

$$\|e(t)\| \leq \frac{a\gamma}{b}\|x(t)\|, \tag{4}$$

where: $\|\cdot\|$ is Euclidean norm, $a = \lambda_{min}(Q), \quad b = 2\|PBK\|$ and P is obtained by solving the Lyapunov equation

$$A_{cs}^T P + PA_{cs} = -Q, \tag{5}$$

where $A_{cs} = A + BK$ describes a closed-loop system in the absence of $e(t)$ and Q is chosen such that $Q = Q^T > 0$. Using $\sigma = \frac{a\gamma}{b}$ with $\gamma < 1$ the *standard event-generation rule* can be stated as:

$$\|e(t)\| \geq \sigma\|x(t)\|, \tag{6}$$

which renders (3) asymptotically stable. Condition (6) defines an event-triggered implementation that consists of continuously checking (6) and triggering a new calculation of the control input whenever the inequality becomes valid.

A. Problem Formulation

In this paper we aim to adapt and extend the event-generation rule (6) to solve the step reference tracking problem. More general reference tracking functions are left for future work. The problems arising when performing event-triggered control for reference tracking is further discussed in Section IV. Let us consider the general tracking linear system:

$$\dot{x}(t) = Ax(t) + Bu(t) + Gr(t), \quad y(t) = Cx(t), \tag{7}$$

where $x(t) \in \mathbb{R}^n$, $u(t) \in \mathbb{R}^m$, $y(t) \in \mathbb{R}^p$ and $r(t) \in \mathbb{R}^{\acute{p}}$ where $\acute{p}$ is equal to the number of outputs to be tracked. Now using (2) in the system (7) we achieve the following closed-loop system under a sample and hold implementation of the controller $u(t)$. Similar to (3), we can write (7) including measurement error $e(t)$ as follows:

$$\dot{x}(t) = (A + BK)x + BKe(t) + Gr(t). \tag{8}$$

To show (8) is ISS with respect to $e(t)$, we will eliminate $r(t)$ from (8) and hence transform it to the form similar to (3). In our case, the step reference tracking problem has been solved in following three steps:

Step 1: Formulate Tracking Control Problem: Looking at (6) we can see that the event-generation rule involves only the states of the plant and it will execute either for non-zero initial condition or for disturbance at plant input. So the standard event-generation rule (6) suffers from a *self-start problem* and hence it cannot work for reference tracking. The solution is to incorporate the reference signal $r(t)$ into the event-generation rule (6). However, we cannot simply add $r(t)$ to the rule since a reset mechanism is also required to make $e(t)$ zero again after an event generation. The simple idea is to make $r(t)$ part of the state by augmenting the system (1) with an integrator state as $x_i(t) = \int_0^t (r(t) - C_i x(t))dt$, where $x_i(t) \in \mathbb{R}^{\acute{p}}$ whereas the matrix C_i has appropriate order i.e. $(\acute{p} \times n)$.

Now the system (1) can be re-written on the form of tracking system (7) as follows:

$$\underbrace{\begin{pmatrix}\dot{x}(t)\\ \dot{x}_i(t)\end{pmatrix}}_{\dot{x}_a(t)} = \underbrace{\begin{pmatrix}A & \mathbf{0}\\ -C_i & \mathbf{0}\end{pmatrix}}_{\bar{A}}\underbrace{\begin{pmatrix}x(t)\\ x_i(t)\end{pmatrix}}_{x_a(t)} + \underbrace{\begin{pmatrix}B\\ \mathbf{0}\end{pmatrix}}_{\bar{B}}u(t) + \underbrace{\begin{pmatrix}\mathbf{0}\\ \mathbf{I}\end{pmatrix}}_{\bar{G}}r(t),$$
$$\dot{x}_a(t) = \bar{A}x_a(t) + \bar{B}u(t) + \bar{G}r(t), \tag{9}$$

where $x_a(t) \in \mathbb{R}^{n+\acute{p}}$ and $\bar{A}, \bar{B}$ and $\bar{G}$ are $(n+\acute{p}) \times (n+\acute{p}), (n+\acute{p}) \times m$ and $(n+\acute{p}) \times \acute{p}$ matrices respectively.

Step 2: Shift Origin of the System: Since we are aiming to asymptotically track a step reference signal, problem (9) can be formulated as a stabilization/regulation problem to a non-zero origin. Here we first shift the origin of the system to the desired equilibrium point using a change of variables. Let us introduce the desired state vector $x_a^*(t)$ as the equilibrium point. The control required to keep the crane at $x_a^*(t)$ is computed by setting $\dot{x}_a(t) = 0$ in (9) and then solving for $u^*(t)$:

$$\bar{A}x_a^*(t) + \bar{B}u^*(t) + \bar{G}r(t) = 0 \tag{10}$$
$$u^*(t) = -\bar{B}^+\left(\bar{A}x_a^*(t) + \bar{G}r(t)\right) \tag{11}$$

where $\bar{B}^+ = \left(\bar{B}^T\bar{B}\right)^{-1}\bar{B}^T$ is a pseudoinverse for a case when $(n+\acute{p}) > m$. $u^*(t)$ would exist if $\bar{B}$ has full column rank. Now, we introduce the new variables:

$$\xi(t) = x_a(t) - x_a^*(t), \quad \tilde{u}(t) = u(t) - u^*(t). \tag{12}$$

Since $x_a^*(t)$ is fixed, the dynamics of new state $\xi(t)$ is given by $\dot{\xi}(t) = \dot{x}_a(t)$ so, by differentiating (12), we achieve:

$$\dot{\xi}(t) = \bar{A}x_a(t) + \bar{B}u(t) + \bar{G}r(t). \tag{13}$$

Inserting $x_a(t) = \xi(t) + x_a^*(t)$ and $u(t) = \tilde{u}(t) + u^*(t)$ in (13),

$$\dot{\xi}(t) = \bar{A}\xi(t) + \bar{B}\tilde{u}(t) + \underbrace{\bar{A}x_a^*(t) + \bar{B}u^*(t) + \bar{G}r(t)}_{=0}, \quad (14)$$

we achieve a new system with the origin shifted to $x_a^*(t)$:

$$\dot{\xi}(t) = \bar{A}\xi(t) + \bar{B}\tilde{u}(t) \quad (15)$$

Step 3: Apply Lyapunov ISS theorem: By performing a change of variables we have transformed the problem of asymptotically tracking a step reference into a regulation problem similar to (1) but with non-zero origin. Now, we can apply the Lyapunov ISS theorem to (15). Since the reference signal is a step, there will be no measurement error in the reference signal, and so, the measurement error can be written as:

$$e(t) = x_a(t_k) - x_a(t) \quad (16)$$

achieving the closed-loop system under measurement error as:

$$\dot{\xi}(t) = \left(\bar{A} + \bar{B}K\right)\xi(t) + \bar{B}Ke(t) \quad (17)$$

Following the same procedure shown in the previous section the *extended event-generation rule* for asymptotically tracking a step reference signal, can be stated as follows:

$$\|e(t)\| \geq \sigma \|\xi(t)\| = \sigma \|x_a(t) - x_a^*(t)\|, \quad (18)$$

where $\sigma = \frac{a\gamma}{b}$, and $\gamma < 1$, which renders (17) asymptotically stable. Condition (18) defines an event-triggered implementation that consists of continuously checking (18) and triggering a new calculation of the control input whenever the inequality becomes valid i.e. $\|e(t)\|$ becomes greater than or equal to $\sigma\|\xi(t)\|$.

In the following section we introduce the model of a 3D tower crane lab process and the control design approaches for performing event-triggered and periodic control of such a system.

III. 3D Tower Crane Modeling and Control Design

We start by giving a general mathematical model and description of the 3D tower crane, followed by the presentation of a simplified system model where we split the full system into two subsystems. Using a Linear Parameter Varying (LPV) description of the system model, we are able to employ linear control design techniques and achieve a stabilizing controller for the lab process.

A. Mathematical Model of the 3D Tower Crane

The lab model of the 3D tower crane shown in Fig. 1 is an 8th[1] order multi-input multi-output (MIMO) system with highly nonlinear dynamics and strong coupling between various inputs and outputs [13]. A full system identification of the lab process is done in [14]. The 3D tower crane is modeled by the following coupled first-order ordinary differential equations (for a detailed description see [14]):

$$\dot{x}(t) = f(x(t), u(t), L), \quad y(t) = Cx(t), \quad (19)$$

[1]Here we are not considering the z-axis actuator dynamics that is the model for load lift-line motor. The lift-line length L will be considered constant for each step command but it can be varied after achieving equilibrium and before changing to next step level.

where $x(t) \in \mathbb{R}^8$, $u(t) \in \mathbb{R}^2$ and $y \in \mathbb{R}^4$. Instead of designing one centralized controller for the whole dynamical system we propose to split the system into two simpler subsystems and design the controller for each, independently. In [14] the tower crane system has been split into two subsystems namely *Trolley Subsystem* and *Arm Subsystem*. Note that these subsystems are coupled only by $x_w(t)$ (Trolley Position). Furthermore, note that the controller for each subsystem can be designed separately and also it can use only the local state information. Using L and $x_w(t)$ as parameters, we can transform the nonlinear system into LPV subsystems as given below:

*1) **Trolley Subsystem**:*

$$\begin{aligned} \dot{x}_1(t) &= A_1(L)x_1(t) + B_1(L)u_1(t) \\ y_1(t) &= C_1 x_1(t), \end{aligned} \quad (20)$$

where $x_1(t) = [x_w(t), \dot{x}_w(t), \alpha(t), \dot{\alpha}(t)]^T$ and $y_1(t) = [x_w(t), \alpha(t)]^T$. Also note that $u_1(t) \in \mathbb{R}$ is a control signal for trolley motor, $x_w(t)$ is a trolley position, $\alpha(t)$ is an in-plane payload angle and $L \in [0, 1]\,m$ is the length of lift-line. Note that the *Trolley Subsystem* (20) is linear and parameterized in terms of L.

*2) **Arm Subsystem**:* Since the only coupling term between the arm and trolley subsystems is $x_w(t)$, we treat the trolley position $x_w(t)$ and lift line length L as the varying parameters inside the arm subsystem as follows:

$$\begin{aligned} \dot{x}_2(t) &= A_2(x_w(t), L)x_2(t) + B_2(x_w(t), L)u_2(t) \\ y_2(t) &= C_2 x_2(t), \end{aligned} \quad (21)$$

where $x_2(t) = [\theta(t), \dot{\theta}(t), \beta(t), \dot{\beta}(t)]^T$ and $y_2(t) = [\theta(t), \beta(t)]^T$. Note that $u_2(t) \in \mathbb{R}$ is a control signal for arm motor, $\theta(t)$ is an arm position and $\beta(t)$ is an out-of-plane payload angle. Note that the system matrices of the *Arm Subsystem* (21) are parameterized in terms of L and $x_w(t)$. Also note that (20) and (21) are an LPV description of the original highly nonlinear system (19). The nonlinearities are hidden inside matrices $A_1(L)$, $A_2(x_w(t), L)$, $B_1(L)$ and $B_2(x_w(t), L)$. Now we are able to use linear controller design methods for our nonlinear system. For general details of LPV design methods see [15]. We are now ready to design the controllers which are able to stabilize the tower crane system.

B. Full State Feedback Integral Control and State Estimation

Two controllers K_1 and K_2, have been designed separately on the basis of (20) and (21) for the *Trolley Subsystem* and *Arm Subsystem*, respectively. Due to space limitations we only summarize the controller design proposed in [14].

A pole-placement method is used for the design of the controller with both Gain-Scheduling and Fixed-Gain techniques [16]. Even though gain-scheduling controllers would achieve higher control performance, its implementation poses a high computation overhead due to several floating point multiplications. Hence, to save the controller computational resources, a fixed-gain controller has been devised by carefully selecting the equilibrium point $(x_w(t), L)$. For a fixed length of $L = 0.16$, it is shown in [14] that for the whole range of points $(x_w(t), L)$ the designed controllers render the

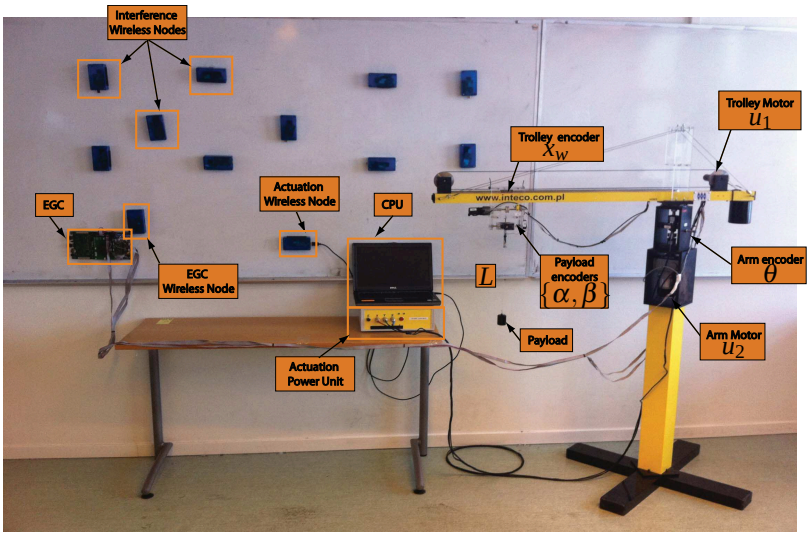


Fig. 1. Experimental setup for performing event-triggered control of 3D tower crane lab process [13]. The system is composed of a 3D tower crane, an Event-Triggered Generator (EGC) and several nodes communicating over the IEEE 802.15.4 wireless network

closed-loop system (7) stable. For stability evaluation we assume that the coupling parameter $x_w(t)$ varies slowly, such that the LPV systems (20) and (21) are valid [15].

For the estimation of the full states $x_1(t)$ and $x_2(t)$ for each of the subsystems, a discrete-time first-order approximation of the nonlinear model (19) was made. This approach provided very accurate state estimation which achieved good control performance, as it will be shown in Section V.

C. Event-Triggered Control (ETC)

Here we present the ETC design methodology presented in Section II-A to the specific case of the 3D tower crane model given in the previous section. For this purpose we linearize the two subsystems (20) and (21) around $(x_w, L) = (0.44, 0.16)$ and then augment each with integrator states $x_{i1} = \int_0^t (r_1(t) - x_w(t))dt$ and $x_{i2} = \int_0^t (r_2(t) - \theta(t))dt$ respectively and then combine them to achieve the complete state-space system.

Using the complete state-space system we first calculate the matrix P for the closed-loop system by setting $Q = \mathbf{I}$ in the Lyapunov equation (5). Note that we may use any $Q = Q^T > 0$. Then, using the computed P, Q, for $A = \begin{pmatrix} A_1 & \mathbf{0} \\ \mathbf{0} & A_2 \end{pmatrix}$, $B = \begin{pmatrix} B_1 & \mathbf{0} \\ \mathbf{0} & B_2 \end{pmatrix}$, and $K = \begin{pmatrix} K_1 & \mathbf{0} \\ \mathbf{0} & K_2 \end{pmatrix}$, and $\gamma = 1$, the extended event-generation rule (18) becomes

$$\|e(t)\| \geq 0.008 \|\xi(t)\|. \tag{22}$$

Where $\bar{A}_1, \bar{A}_2, \bar{B}_1$ and $\bar{B}_2$ have the same form as given in (9).

D. Periodic Time-Triggered Control (TTC)

The periodic TTC is designed to allow for a control performance comparison with the proposed ETC design. Several tests were conducted to evaluate the control performance for different sampling period values ranging from 20ms to 100ms. It was seen that a sampling period of $T_s = 40$ms is the greatest value with no significant effect in the control performance of the 3D tower crane. Therefore, we choose this value as the sampling period when implementing the periodic controller.

Remark 1: The procedure to choose the sampling period T_s for the comparison of a periodic TTC and ETC designs is not trivial. A natural choice would be to select T_s as maximum sampling time for which the closed-loop system remains stable for any possible initial condition, as proposed in [9]. According to [5], one can use for T_s the minimum inter-sampling time t_{min} given by the event-generation rule (18). However, it is noted in [9] that t_{min} achieved in [5] is rather conservative, and so, the authors propose another method for the calculation of tighter bounds on t_{min}. For the tower crane system presented in Section III-A we achieved $t_{min} = 0.1$ms following [5] and verified that $t_{min} = 25$ms using the methodology in [9] (when aiming for the highest control performance). Another possible choice is to practically verify what value of T_s allows for a certain level of performance of the closed-loop system under many possible working conditions. The drawback of the former approach is that it might be conservative in a real evaluation for the given working conditions. The latter choice might not be feasible since several tests need to be produced to compare the control performance under different values of T_s. However, in this paper we chose the later approach since we want to clearly show in practice, the merits of an ETC design.

IV. Experimental Setup

In this section we first describe the experimental setup required to perform event-triggered control of dynamical systems as proposed in Section II and III-C. We propose a dedicated circuit for implementing low-power event-triggered controllers in networked control systems. Implementation issues when designing an event-triggered controller for a real control system are also discussed.

A. Overview

The experimental setup is composed of three main components: the 3D tower crane, an event-generation circuit (EGC) and the wireless network. Fig. 1 shows all the components of the proposed networked control system. The EGC is

responsible for performing all the sensor readings, estimating the full states given by (20) and (21) and implementing the event-generation rule (22). In this setup we chose to perform the calculation of the control inputs $u(t_k)$ in the EGC for simplicity of analysis of the obtained results. When an event-generation rule (22) becomes valid at the EGC, the control signals are computed and transmitted to the *EGC wireless node*, which then transmits a data packet with $u(t_k)$ over the IEEE 802.15.4 wireless network to the actuation wireless node. The *Actuation wireless node* is connected to a CPU running LabVIEW [17], which transmits the control signals to an actuation circuitry, which actuates in the 3D tower crane arm and trolley motors.

B. Wireless Network

The wireless network is responsible for providing a communication link between the EGC and the CPU that performs the actuation on the tower crane. The network is composed of two Telos wireless sensor nodes [18], one connected to the EGC through a serial interface which we denote by *EGC wireless node* and another connected to the CPU, which we denote by *Actuation wireless node*. Both wireless nodes are event-driven in the sense that at every time a new control input is generated by the EGC, the EGC wireless node will be requested to transmit a packet with the control inputs data. Moreover, when new data arrives at the Actuation wireless node, it will automatically report it to the CPU and request LabVIEW [17] to perform a new actuation in the 3D tower crane.

These devices communicate using the low-power wireless IEEE 802.15.4 standard for the physical and medium access control (MAC) layers [19]. The MAC scheme used in this experimental setup is the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA), which is a contention-based scheme where each node is able to sense if the channel is busy before transmitting a message. Moreover, the CSMA/CA algorithm allows for retransmissions if a data packet is lost during the transmission (no acknowledgment received after the transmission), or if the wireless channel is busy at the transmission time. A retransmission takes place in an exponential back-off fashion. The maximum number of retransmission $maxR_{tx}$, can be defined by the user. For more details on the protocol, see [19]. The operating system running in each wireless node is TinyOS for which the standard MAC algorithm was developed [20].

In addition to the two wireless devices used for the control of the 3D tower crane, 10 additional *interference nodes* share the same wireless network. In this way we are able to generate heavy traffic conditions which allows us to verify the robustness of the control system under data losses and varying transmission delays.

C. Event-Generation Circuit

The EGC performs four main functions: quadrature decoding of encoders, state estimation, evaluation of event-generation rule and control computation. This circuit has been implemented using a dedicated Atmel's ATxmega128A1 MCU (micro-controller unit). The selected MCU is a very powerful $8/16$ bit device which can give a throughput up to 32MIPS (mega instructions per second) and it is a natural choice for an implementation of an event-triggered controller. Adding to this, the MCU has a large range of peripherals available, with a low power consumption and a dedicated event routing network which ensures predictable response time especially for tasks as data acquisition and data transfer.

TABLE I
TIMING SPECIFICATION AND POWER CHARACTERISTICS OF THE EVENT-GENERATION CIRCUIT (EGC)

Parameter Description	**Values**
Encoder Reading	$182\,\mu s$
Speed Estimation	$160\,\mu s$
Event-Generation Rule	$740\,\mu s$
Control Computation	$75.9\,\mu s$
Serial Transmission	$1.120\,ms$
Peripheral Initialization	$216\,\mu s$
System Clock	32 MHz
Operating Voltage	$3V$
Total Active Mode Current	$16.2184\,mA$
Total Idle Mode Current	$0.5840\,mA$
Active Mode Duty Cycle	22.8%
Total Average Current	$4.1486\,mA$

The interface between the quadrature encoders in the tower crane is made by external dedicated chips which transmit the rotation/distance measurements to the EGC.

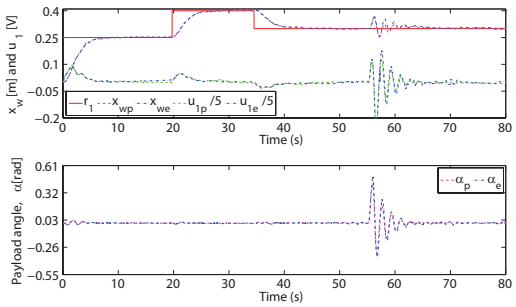
The EGC can be utilized in several configurations: as event-triggered sensor and controller where $u(t_k)$ can be directly applied to a plant or sent to an actuator node or simply as event-triggered sensor where the full state $x(t_k)$ is sent to the controller node. Moreover, this circuit can also be used for self-triggered [9] and periodic time-triggered control.

Other than quadrature encoders, the circuit provides Analog-to-Digital converters (ADC) to read analog sensors and provides Digital-to-Analog converters (DAC) outputs for performing actuation tasks. Interface between the EGC and any other processing unit as a wireless node or a CPU can be made over serial communication.

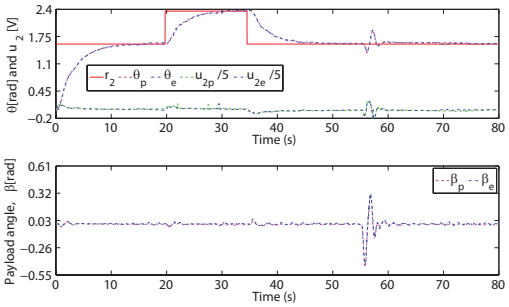
In our current experiment, the EGC reads four encoders of the crane every 10 ms, containing the information of the arm angle rotation $\theta(t)$, trolley position $x_w(t)$ and payload angles $\alpha(t)$ and $\beta(t)$, estimates the full state vector $x_a(t_k)$ and evaluates the event-generation rule (22). If an event is generated, the control signal $u(t_k)$ is sent over a serial bus $@115.2\,kpbs$ to the EGC wireless node which transmits $u(t_k)$ over wireless to the actuation wireless node.

Fig. 1 shows the first prototype of the EGC with quadrature decoding, which has been developed on a two-layer PCB (printed circuit board). In the final version, we aim to implement the EGC on a four-layer PCB to achieve a small form-factor for portable applications as decentralized sensing and control.

We now present a short analysis on the timing and power characteristics of the EGC. As it is natural for a control application, the delays imposed by the EGC for computations should be small. Moreover, since the device is to be portable and deployed wirelessly, operating for several months/years with a battery as its energy source, one has to take into

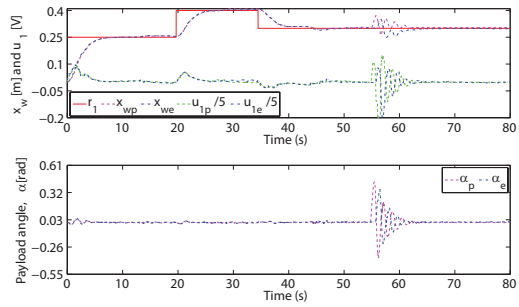


(a) Time response of Trolley subsystem and control input to the trolley motor. Subscript 'p' is for TTC and 'e' for ETC.

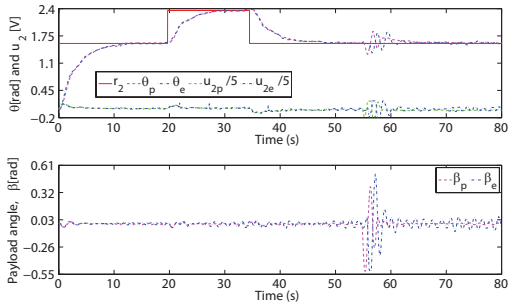


(b) Time response of Arm subsystem and control input to the arm motor.

Fig. 2. Comparison between a periodic time-triggered controller (TTC) and an event-triggered controller (ETC) for step reference tracking with no losses.



(a) Time response of Trolley subsystem and control input to the trolley motor.



(b) Time response of Arm subsystem and control input to the arm motor.

Fig. 3. Comparison between a periodic time-triggered controller (TTC) and an event-triggered controller (ETC) for step reference tracking under heavy losses. Ten additional wireless nodes are deployed in the same network as the 3D tower crane process, transmitting periodic messages every $T = 10$ms.

account that the power consumption should be as small as possible.

1) EGC Timing Characteristics: The computation time required for various functions of the EGC are given in Table I. These timings are given for a 32 MHz clock rate and are true values and not approximates. The total time taken when using the ETC approach varies between $[1.082, 2.228]$ms depending on whether an event occurred or not, whereas the periodic TTC takes a fixed computation time of 1.538 ms during each sampling interval.

2) EGC Power Characteristics: We are running the MCU in two power modes: Active mode and Idle mode. The MCU remains in Idle mode (@2MHz) all the time, except when executing the Interrupt service routine (ISR) for a timer, which overflows every 10ms, and generates an interrupt to bring the MCU into active mode within $0.17\,\mu$s. When the MCU enters in the ISR, it switches the clock source to 32MHz to achieve the maximum throughput for fast processing, hence, significantly decreasing the duration of active mode. Before coming out of the ISR, the MCU switches the clock back to 2MHz to reduce power consumption. In Idle modes all peripherals other than one timer remains disabled while in active mode only the timer and the serial interface are active. By following this scheme we have saved a large amount of power and statistics are given in Table I. Every time the EGC detects an event, it will trigger a transmission of a message by the wireless node which is an extremely battery consuming task, with an average current consumption of 20mA per transmission [18]. On the other hand, when the wireless node is in idle mode it consumes 40uA. Hence, the number of message transmissions/events must be minimized in order to increase the device's lifetime.

D. Implementation Issues

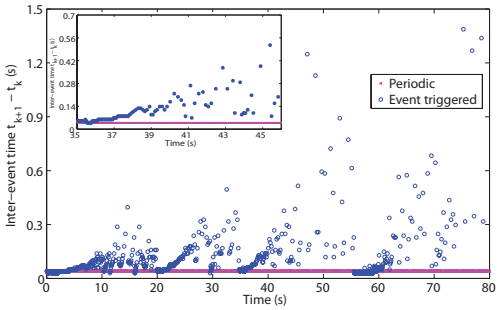
When implementing event-triggered control in a real dynamical system several issues may arise. Here we explore the problems when performing integral control and quantization effects on the event-generation rule.

1) Integral Control: In order to perform tracking of a reference signal with no steady state errors, an integral dynamic controller must be implemented. Since the sampling period is varying in the event-triggered controller case, errors when performing the computation of the integral would arise.

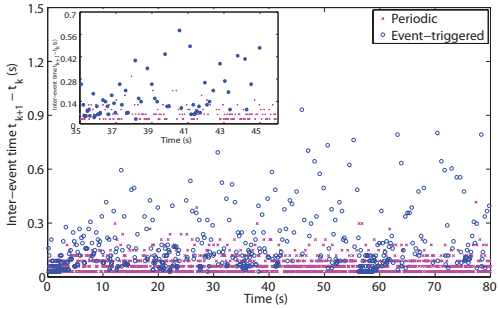
Since there are no stability proofs on the use of event-triggered control with dynamic controllers as Proportional-Integral (PI) controllers, we solve this issue by treating the integrator states $x_i(t_k)$ as a state of the plant. In the case of the EGC not being collocated with the controller unit, we can compute $x_i(t_k)$ periodically on the EGC, and transmit this value to the controller. It is necessary for the reference signal $r(t)$ to be sent to the EGC as well as the controller, which increases the communication costs.

2) Quantization Effect at Equilibrium: When the system converges to the desired reference signal, the numerical values of both $e(t)$ and $\xi(t)$ become ≈ 0. Therefore, due to quantization effects, false or undesired triggering may occur. To address these issues a small offset δ is added to (22). Incorporating all the changes, an *Improved Event-Generation Rule* is given by:

$$\|e(t)\| \geq 0.008\,\|\xi\| + \delta \tag{23}$$



(a) Inter-event times $t_{k+1} - t_k$ with no packet losses ($N_L = 0$) and no external nodes ($M_{nodes} = 0$).



(b) Inter-event times $t_{k+1} - t_k$ with packet losses ($N_L = X$) and 10 external nodes ($M_{nodes} = 10$).

Fig. 4. Inter-event times $t_{k+1} - t_k$ for a periodic time-triggered controller (TTC) and an event-triggered controller (ETC) for step reference tracking with and without the influence of packet losses.

For our implementation we chose a parameter $\delta = 0.005$.

V. Results

In this section we show experimental results on the tracking of step reference functions by the 3D tower crane lab process under both a periodic TTC and ETC policies. We consider the case where the trolley position $x_w(t)$ and the arm angle $\theta(t)$ should follow given reference signals $r_1(t)$ and $r_2(t)$, respectively. At the same time the trolley position and arm angle are being tracked, it is our goal that the payload angles $(\alpha(t), \beta(t))$ remain as small as possible and quickly approaching a zero angle value.

The reference signals $r_1(t)$ and $r_2(t)$ selected for experiment are stepwise functions given by:

$$r_1(t) = 0.25\,s(t) + 0.15\,s(t-20) - 0.10\,s(t-35) \quad (24)$$

$$r_2(t) = \frac{\pi}{2}\,s(t) + \frac{\pi}{4}\,s(t-20) - \frac{\pi}{4}\,s(t-35) \quad (25)$$

where $s(t-t_i)$ is a unit step function which attains value 1 for $t \geq t_i$.

We will be analyzing the cases when: *A*) Periodic TTC and ETC are not influenced by any packet loss in the wireless channel and *B*) Periodic TTC and ETC are influenced by heavy external traffic caused by 10 additional interference nodes sharing the same network. For the latter case, we are interested in analyzing the influence of packet loss in the performance of both controller designs. Moreover, we provide analysis on the influence of the MAC parameters in the control performance. All the results on packet loss and delay were evaluated by performing several runs of the same experiment.

We denote the sampling period of the TTC approach as T_s, the number of external nodes as M_{nodes}, number of samples N_S, packet loss probability ρ and average transmission delay Δ_t. The MAC parameter we design is the maximum number of retransmissions $maxR_{tx}$. The criterion for comparing the performance of the different sampling schemes is the root mean square (RMS) of the output error defined as

$$\mathbf{RMS} = \sqrt{\frac{1}{N_r}\sum_{i=1}^{N_r} \|z_i(t) - r_i(t)\|} \quad (26)$$

where the state $z(t) = [x_w(t), \theta(t), \alpha(t), \beta(t)]^T$, $r(t) = [r_1(t), r_2(t), 0, 0]^T$ and $N_r = T_r/\Delta_r$ is the number of samples taken of $z(t)$ in LabVIEW with a period $\Delta_r = 5$ms for a simulation length of $T_r = 50$s.

A. Periodic TTC and ETC Comparison

Here, we compare both periodic TTC and ETC approaches over a reliable wireless channel where no losses occur. Evaluating the event-generation rule (23) for the proposed experimental setup, we achieve a minimum inter-event $t_{min} = 30$ms. Fig. 2(a) shows the response of trolley subsystem and Fig. 2(b) shows the response of the arm subsystem. Moreover, at time $t = 55$s a disturbance is introduced that excites both payload angles. We can see that the response for both TTC and ETC are very similar and no major differences are visible. Table II shows the statistics for both control policies. Both controllers have rejected the disturbance very quickly. Moreover, we can see that the RMS values for both ETC and TTC are very close. In Fig. 4(a), the inter-event times $t_{k+1} - t_k$ are presented. For the evaluated case, the event-generation rule (23) provides inter-event times that behave approximately exponentially according to how far the system is from the reference point. It is clear that the ETC has guaranteed the same level of control performance of the TTC, but using 61% less controller updates. Thus, the network utilization and battery consumption is extremely reduced by using the proposed event-triggered scheme.

B. Packet Losses and Effect of MAC

After providing results on the performance of both control schemes when performing reference tracking and rejecting disturbances, we discuss now how packet losses and varying transmission delays affect control performance. Moreover, we evaluate the influence of the choice of the maximum number of allowed retransmissions.

Every time (23) is true, the measurement error $e(t)$ is reset to zero. In this way, even though the control input may be lost in the wireless transmission, the EGC will not generate a new event straight away. The EGC wireless node is responsible for retransmitting the control input to the actuator wireless node if $maxR_{tx} \neq 0$.

In this case, 10 interference nodes ($M_{nodes} = 10$) are sharing the same network, periodically transmitting a message every 10ms. Fig. 3 shows the time response of the trolley and arm subsystems as well as the control inputs given to each subsystem when $maxR_{tx} = 0$. The total number

TABLE II

EXPERIMENTAL RESULTS: PERFORMANCE EVALUATION AND STATISTICS OF ETC AND TTC FOR THE SEVERAL TEST CASES, WITH AND WITHOUT PACKET LOSSES.

	T_s	M_{nodes}	$maxR_{tx}$	N_S	ρ	Δ_t	**RMS**
ETC	–	0	0	763	0	5.72	0.362
TTC	40 ms	0	0	2000	0	5.72	0.364
ETC	–	10	0	944	47.2%	5.72	0.371
TTC	40 ms	10	0	2000	46.7%	5.72	0.379
ETC	–	10	3	798	4.7%	9.14	0.368
TTC	40 ms	10	3	2000	4.5%	9.32	0.381

of transmission losses was on average 47%. Again, it is observed that the performance of the ETC is very similar to the TTC scheme. In a closer analysis, one can see that the ETC has a faster rise time and settling time than the TTC. However, when looking at the payload angles, it is noticeable that the oscillations are higher for ETC than TTC, even though this value is extremely small. From the RMS values in Table II, one can verify that the ETC exhibits a slightly better control performance when compared to the TTC. Naturally, the performance with losses is worse than without losses, for both ETC and TTC schemes.

Fig. 4(b) shows the inter-event times for both ETC and TTC schemes. Due to the packet losses, the event-generation rule is triggered more often than the experiments without losses. In this case, the ETC policy generates 47.2% of the samples of the TTC.

We now present the results for the same analysis shown above, with $M_{nodes} = 10$ but $maxR_{tx} = 3$. In this case, the EGC wireless node is able to retransmit a packet up to 3 times.

As presented in Table II, we see that by allowing retransmissions, the packet losses are reduced from 48% to 4.7%, while the delay Δ_t is increased from 5.72ms to 9.14ms and 9.32ms for the ETC and TTC schemes, respectively. With this increase of delay, the control performance is better than the case with losses. This shows the benefit of allowing retransmissions in networked control systems and demonstrate that the ETC is able to guarantee a good performance under packet losses and delays.

VI. CONCLUSIONS AND FUTURE WORK

The present work provided an experimental illustration of the benefits of performing event-triggered control versus the traditional periodic paradigm, both in terms of energy and bandwidth savings for WSANs applications. We developed and evaluated the first prototype of an event-generation circuit (EGC) for providing a mean of implementing event-triggered control in WSANs. We have experimentally validated the EGC for performing event-triggered control of a real 3D tower crane lab process. From a control point of view, we have evaluated an event-triggered controller against a periodic time-triggered controller, under the effect of disturbances, packet losses and delays. Experimental results show that the event-triggered controller achieves the same level of control performance as compared to a time-triggered controller but with far less communication cost under any of the evaluated conditions.

As future work, we intend to provide stability guarantees in the use of dynamic controllers for event-triggered sampling. Moreover, we plan to extend the approach presented in this paper for the case of general reference tracking. In this paper, the event-triggered controller has been implemented in a centralized fashion. The extension to a decentralized event-triggered control for the 3D tower crane is currently being studied by the authors.

VII. ACKNOWLEDGMENTS

The authors would like to thank Dr. Maben Rabi, Dr. Adolfo Anta, Dr. Manuel Mazo Jr. and Prof. Paulo Tabuada for all the positive discussions while developing this work.

REFERENCES

[1] A. Willig, "Recent and emerging topics in wireless industrial communications: A selection," *IEEE Transactions on Industrial Informatics*, vol. 4, no. 2, pp. 102 –124, May 2008.

[2] P. Antsaklis and J. Baillieul, "Technology of networked control systems," *Proceedings of the IEEE*, vol. 95, no. 1, 2007.

[3] K. Årzén, "A simple event-based PID controller," *Preprints 14th World Congress of IFAC. Beijing, China*, 1999.

[4] K. Åström and B. Bernhardsson, "Comparison of Riemann and Lebesgue sampling for first order stochastic systems," *Proceedings of the 41st IEEE Conference on Decision and Control*, vol. 2, 2002.

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

[6] W. Heemels, J. Sandee, and P. van den Bosch, "Analysis of event-driven controllers for linear systems," *Int. J. of Control*, pp. 81(4), 571–590, 2008.

[7] M. Velasco, J. Fuertes, and P. Martí, "The self triggered task model for real-time control systems," *24th IEEE Real-Time Systems Symposium (work in progress)*, pp. 67–70, 2003.

[8] X. Wang and M. Lemmon, "Self-triggered feedback control systems with finite-gain l2 stability," *IEEE Transactions on Automatic Control*, vol. 45, pp. 452–467, 2009.

[9] M. Mazo Jr, A. Anta, and P. Tabuada, "An ISS self-triggered implementation of linear controllers," *Automatica*, vol. 46, no. 8, pp. 1310–1314, 2010.

[10] D. Lehmann and J. Lunze, "Extension and experimental evaluation of an event-based state-feedback approach," *Control Engineering Practice*, vol. 19, no. 2, pp. 101 – 112, 2011.

[11] M. Rabi and K. H. Johansson, "Scheduling Packets for Event-Triggered Control," *Proceedings of 10th European Control Conference*, pp. 3779 – 3784, Aug. 2009.

[12] A. Cervin and T. Henningsson, "Scheduling of event-triggered controllers on a shared network," in *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, 2008, pp. 3601 –3606.

[13] *Inteco 3D Tower Crane User Manual*, Inteco. [Online]. Available: http://www.inteco.com.pl/

[14] F. Altaf, "Modeling and event-triggered control of multiple 3d tower cranes over WSNs," Master's thesis, Royal Institute of Technology (KTH), Nov. 2010.

[15] W. Rugh and J. Shamma, "Research on gain scheduling," *Automatica*, vol. 36, no. 10, pp. 1401–1425, 2000.

[16] H. Khalil, *Nonlinear systems*. Prentice Hall, NJ, 2002.

[17] *LabVIEW*, National Instruments. [Online]. Available: http://www.ni.com/labview/

[18] J. Polastre, R. Szewczyk, and D. Culler, "Telos: enabling ultra-low power wireless research," *Information Processing in Sensor Networks 2005. Fourth International Symposium on*, Apr. 2005.

[19] *IEEE 802.15.4 standard: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)*, IEEE, 2006. [Online]. Available: http://www.ieee802.org/15/pub/TG4.html

[20] J. Hauer, "TKN15.4: An IEEE 802.15.4 MAC Implementation for TinyOS 2," *TKN Technical Report Series, Telecommunication Networks Group, Technical University Berlin*, no. TKN-08-003, Mar 2009.