INTERNATIONAL JOURNAL OF ROBUST AND NONLINEAR CONTROL
Int. J. Robust Nonlinear Control 2010; **20**:209–225
Published online 19 May 2009 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/rnc.1451

Hybrid model predictive control based on wireless sensor feedback: An experimental study

Alberto Bemporad[1], Stefano Di Cairano[1,2,*,†,‡], Erik Henriksson[3] and Karl Henrik Johansson[3]

[1]*Department of Information Engineering, School of Engineering, University of Siena, 53100 Siena, Italy*
[2]*Powertrain Controls R and A, Ford Motor Company, 2101 Village Road MD 2036, Dearborn, MI 48124, U.S.A.*
[3]*ACCESS Linnaeus Centre, School of Electrical Engineering, Royal Institute of Technology, 10044 Stockholm, Sweden*

SUMMARY

Design and experimental validation of model predictive control (MPC) of a hybrid dynamical laboratory process with wireless sensors is presented. The laboratory process consists of four infrared lamps, controlled in pairs by two on/off switches, and of a transport belt, where moving parts equipped with wireless sensors are heated by the lamps. The process, which is motivated by heating processes in the plastic and printing industry, presents interesting hybrid dynamics. By approximating the stationary heat spatial distribution as a piecewise affine function of the position along the belt, the resulting plant model is a hybrid dynamical system. The control architecture is based on the reference governor approach: the process is actuated by a local controller, while a hybrid MPC algorithm running on a remote base station sends optimal belt velocity setpoints and lamp on/off commands over a wireless link, exploiting the sensor information received through the wireless network. A discrete-time hybrid model of the process is used for the hybrid MPC algorithm and for the state estimator. The physical modelling of the process and the hybrid MPC algorithm are presented in detail, together with the hardware and software architectures. The experimental results show that the presented theoretical framework is well suited for control of the new laboratory process, and that the process can be used as a prototype system for evaluating hybrid and networked control strategies. Copyright © 2009 John Wiley & Sons, Ltd.

Received 18 May 2008; Revised 13 November 2008; Accepted 16 February 2009

KEY WORDS: model predictive control; hybrid dynamical systems; wireless sensor networks; experimental process

1. INTRODUCTION

Today's society is rapidly moving toward an 'everywhere connected wireless community' with large

*Correspondence to: Stefano Di Cairano, 2101 Village Road MD 2036, Dearborn, MI 48124, U.S.A.
†E-mail: sdicaira@ford.com
‡The research described in this paper was accomplished when Dr Di Cairano was with the Department of Information Engineering, School of Engineering, University of Siena.

Contract/grant sponsor: HYCON Network of Excellence, of the European Commission; contract/grant number: FP6-IST-511368

Contract/grant sponsor: Italian Ministry for University and Research (MIUR)
Contract/grant sponsor: European Commission
Contract/grant sponsor: Swedish Research Council
Contract/grant sponsor: Swedish Governmental Agency for Innovation Systems
Contract/grant sponsor: Swedish Foundation for Strategic Research

numbers of interacting mobile embedded systems that influence every aspect of our lives. Handheld communication devices and personal computers are exchanging information over cellular and wireless local area networks. Wireless appliances and systems are being used in our offices and homes, but more and more also in large-scale control systems such as in utility infrastructures and transportation networks.

Communication networks have been commonly used in distributed control systems since the seventies [1]. Over the last few years, a number of initiatives have been taken to introduce wireless networks in industrial automation and process control. New wireless sensors for industrial control applications are already present on the market. It is evident that new communication protocols and control strategies are needed for these wireless control systems [2, 3]. Recent standardization efforts include WirelessHART [4], which is a wireless mesh network communications protocol designed to meet the needs for process automation applications, and the ISA-SP100 standard [5], according to which wireless sensors will be primarily used within 'Class 2-Closed-loop supervisory control' of the taxonomy introduced by the standard, where information availability is often not safety critical. A rigorous framework for designing integrated control and communication systems is however lacking. There are several initiatives to counteract this fact; one such initiative is SOCRADES [6] driven by the European automation industry to develop a design platform for wireless automation.

The use of a wireless technology in feedback control loops raises new challenges. The network medium introduces uncertainties on packets loss, communication outages, transmission delay etc. The impact of these uncertainties on the closed-loop control system depends on many system aspects. For example, some communication protocols guarantee the delivery of a message, but on the other hand give high delay variability. Other protocols provide a less reliable communication, but ensure better delay characteristics. For wireless networks, packet losses typically vary heavily with the radio conditions, hence if the environment is changing or the nodes are mobile, the control system needs to handle the varying network conditions.

1.1. Wireless control in industry

There are several benefits from introducing wireless networking in industrial control applications in general. They can be summarized as follows [6, 7]:

- *Cost*: Wireless links lead to reduced wiring, which constitutes a substantial part of the development cost for many industrial plants simply due to the high price of copper wires. Wireless technology also has the potential to reduce the installation cost, since hardware installation for a wireless network is limited to some routers and gateways.
- *Flexibility*: With wireless links there are fewer physical design limitations, and it is easier to move the equipment. Thanks to mobility and fast reconfiguration, new and better designs can be exploited in the system development and operation.
- *Fault handling*: Connectors and wires lead to many faults in industrial control systems, partly because of cable wear and tear. Consequently, wireless devices have a potential to reduce the downtime for these systems.

There are numerous barriers against wireless networking in industrial control. In a recent survey [8], the main concerns were security and reliability, followed by the fact that there is too little knowledge available and too few industrial products. By developing control architectures and algorithms especially suitable for unreliable wireless links, one can target the reliability issue. By presenting a new pedagogical laboratory process on wireless control, we are in this paper contributing to the awareness of networked control potentials and limitations.

1.2. Main contribution

The main contribution of this paper is to present the design and the experimental validation of hybrid model predictive control (MPC) over wireless links on a new laboratory process built for this purpose at the University of Siena, see Figure 1. The process consists of a transport belt where moving parts equipped with wireless sensors are heated by four infrared lamps. The latter are commanded in pairs by two on/off switches. The process is actuated by a local controller, while a hybrid MPC algorithm running on a remote

DOI: 10.1002/rnc

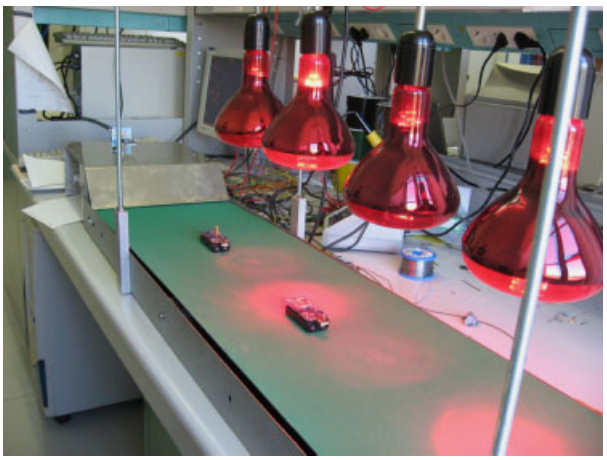

Figure 1. The new hybrid laboratory process with wireless sensors presented in the paper.

base station sends optimal belt velocity setpoints and lamp on/off commands. The remote controller receives information from the sensors through a wireless network formed by Telos motes [9]. A discrete-time hybrid model of the process is used for the design of the hybrid MPC algorithm and of the state estimator.

1.3. Related work

Control over wireless networks is a young research area without mature theory or tools, but with a lot of current activity. A general view on the need for interaction between control and communication in the design of wireless networks was recently introduced [2]. Open research problems in the area of control using wireless sensor networks (WSNs) include choice of architectures and modular design and implementation [3, 10]. A cross-layer framework for the joint design of wireless networks and distributed controllers is being attempted [11], although care needs to be taken to avoid undesirable interactions [12].

This paper advocates the use of MPC [13] as a tool to tackle control problems in such uncertain environments such as those arising from wireless sensors and actuator loops. MPC is widely spread in the industry for the control of complex multivariable processes [14]. Given a model of the process dynamics, constraint specifications on system variables (input saturations, state bounds, etc.), and desired performance specifications, at each time step the MPC control algorithm solves an optimal control problem, which depends on the current state as initial condition and on the current reference signals, over a future prediction horizon. The result of the optimization is a sequence of future control moves. Only the first element of the sequence is applied to the process, the remaining moves are discarded, and the optimization is repeated at the control cycle.

MPC based on *hybrid* dynamical models [15] has emerged as a very promising approach to handle switching linear dynamics, on/off inputs, logic states, as well as logic constraints on input and state variables. The associated finite-horizon optimal control problem can be formulated as a mixed-integer program for which efficient solvers are available. Extensions of the hybrid MPC formulation introduced above have been recently proposed for stochastic hybrid systems [16] that appear to be suitable for application within a hybrid networked control architecture.

To handle the unreliability of the communication links between the base station (where the MPC computations are performed) and the process, we use the reference governor approach [17, 18]. Here, the process stability is granted by a local controller at the plant. The local controller receives its reference from the remotely executed MPC algorithm, which aims at obtaining the desired performance. Thus, the computational power required to solve the optimization problem is moved away from the plant. The reference governor approach was first studied in the context of unreliable network links in [19], where the command sequences computed by the predictive controller are used to possibly overcome packet loss and large (possibly unbounded) delays, together with a synchronization algorithm. Other existing laboratory facilities for feedback control over wireless measurements include platforms for coordination of mobile agents, see e.g. [20].

1.4. Outline

The outline of the paper is as follows. Section 2 presents the laboratory process and the corresponding

DOI: 10.1002/rnc

mathematical model. Section 3 provides a description of the overall control system architecture, including the wireless networks, and Section 4 focuses on the hybrid MPC system design. The hardware and software architectures are detailed in Section 5. Simulations and experimental results are reported in Section 6. Conclusions and a discussion on future research directions are given in Section 7.

Earlier results related to the work described in this paper were presented in [21], where the problem and the hybrid MPC design were introduced. In this paper we extend such preliminaries in several directions. We define the hybrid MPC problem in more detail, thoroughly describe the control and communication architecture, and present extensive experimental results under different network conditions.

2. PROCESS DESCRIPTION, MODELLING, AND ARCHITECTURE

This section describes the laboratory process and derives a control and estimation-oriented hybrid dynamical model of it.

2.1. Physical plant

The physical plant is shown in Figure 1. The main components of the plant, whose schematics are shown in Figure 2, are a belt with an incremental angular encoder, four infrared lamps, and wireless sensors, also called sensor motes, to be placed on the belt. The belt is actuated by an electric servo-motor to which the encoder is connected. The heating lamps are placed in a row over the belt, and two on/off switches are available to actuate them. The first switch controls lamps 1 and 3, the second switch, lamps 2 and 4. The lamps are grouped to reduce the complexity of the model and of the control algorithm. The sensor motes are radio-equipped temperature sensors, which communicate their reading to a remote receiver.

To derive a dynamic model of the process we define the states as the sensor casing temperature $T_1 \in \mathbb{R}$, the sensor temperature $T_2 \in \mathbb{R}$, and the position $p \in \mathbb{R}$ of the part that moves along the belt. The system evolution is

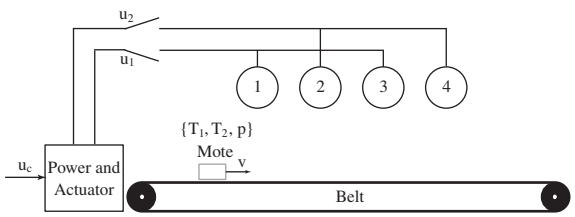

Figure 2. Schematics of the process.

governed by the differential equations

$$\dot{T}_1 = -\alpha(T_1 - T_{\text{ss}}(p, u_1, u_2)) \tag{1a}$$

$$\dot{T}_2 = -\beta(T_2 - T_1) \tag{1b}$$

$$\dot{p} = \gamma(u_c) \tag{1c}$$

where $u_c \in \mathbb{R}$ and $u_1, u_2 \in \{0, 1\}$ are control inputs, and $T_{\text{ss}} : \mathbb{R}^3 \to \mathbb{R}$ is a static nonlinearity. The parameters $\alpha, \beta > 0$ are physical constants, that we have computed by identification on experimental data. The continuous signal $\gamma(u_c)$ corresponds to the part velocity, which is obtained through a static nonlinear mapping $\gamma(\cdot)$ of the control command. As regards the discrete input signals, $u_1 = 0$ when lamps 1 and 3 are off, $u_1 = 1$ when they are on, and similarly for u_2 relatively to lamps 2 and 4. The steady-state temperature of the sensor casing at position p, when the lamps switches are (u_1, u_2), is $T_{\text{ss}}(p, u_1, u_2)$

$$T_{\text{ss}}(p, u_1, u_2) = f_1(p)u_1 + f_2(p)u_2 + T_{\text{amb}} \tag{2}$$

where $T_{\text{amb}} \in \mathbb{R}$ is the ambient temperature, and $f_i(p) : \mathbb{R} \to \mathbb{R}$, $i \in \{1, 2\}$ describe the increase in steady-state temperature at position p obtained by turning on the ith switch.

2.2. Hybrid model

We want to approximate the continuous-time model (1) and the nonlinearity T_{ss} in (2) by a hybrid model. In order to use the tool [22], we introduce an auxiliary variable χ to model a piecewise affine approximation of $(T_{\text{ss}} - T_{\text{amb}})$: we partition $\mathbb{R}$ into ℓ intervals $\{I_1, I_2, \ldots, I_\ell\}$, and approximate f_i, $i = 1, 2$, by the

DOI: 10.1002/rnc

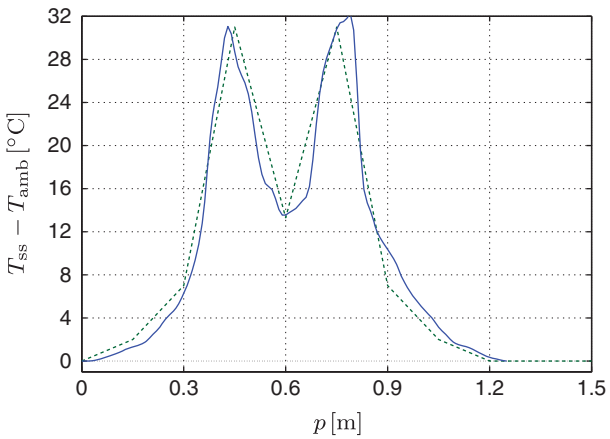

Figure 3. $T_{ss} - T_{amb}$ and its piecewise affine approximation.

piecewise affine functions

$$\chi_i(p(t)) = \begin{cases} K_j^i p(t) + h_j^i & \text{if } u_i = 1 \\ & p \in I_j,\ j = 1, \ldots, \ell \\ 0 & \text{otherwise} \end{cases} \quad i = 1, 2 \tag{3a}$$

$$\chi(p(t)) = \chi_1(p(t)) + \chi_2(p(t)) \tag{3b}$$

The notation $\chi(p(t))$ is used in (3) to highlight that χ depends on the position p, which changes in time. For simplicity, from now on we will use the notation $\chi(t)$ instead. The effect of T_{amb} will be introduced later as a measured disturbance. The nonlinear function and its piecewise affine approximation are shown in Figure 3.

The continuous-time model of the physical plant is sampled with sampling period $T_s = 250$ ms using a zero-order hold. We obtain the following discrete-time system:

$$x(t+1) = \underbrace{\begin{pmatrix} a_{11} & 0 & 0 \\ a_{21} & a_{22} & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{\Phi} x(t) + \underbrace{\begin{pmatrix} b_{11} & 0 \\ b_{21} & 0 \\ 0 & b_{32} \end{pmatrix}}_{\Gamma} \begin{pmatrix} \chi(t) \\ v_c(t) \end{pmatrix} \tag{4a}$$

$$y(t) = \underbrace{\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{C} x(t) \tag{4b}$$

where $x = (T_1, T_2, p)^{\mathrm{T}}$, $\chi(t)$ is defined by (3), and the belt velocity $v_c = \gamma(u_c)$ is used as system input. The motor command can be recovered by the inversion $u_c(t) = \gamma^{-1}(v_c(t))$. By (3)–(4), the quantities $\chi(t)$, $x(t)$, $y(t)$ depend on the discrete inputs $u_1(t)$ and $u_2(t)$.

In order to apply hybrid MPC, systems (3)–(4) are formulated in HYSDEL [22] and automatically converted into a mixed logical dynamical (MLD) system [15]

$$x(t+1) = Ax(t) + B_1 u(t) + B_2 \delta(t) + B_3 z(t) \tag{5a}$$

$$y(t) = Cx(t) \tag{5b}$$

$$E_2 \delta(t) + E_3 z(k) \leqslant E_1 u(k) + E_4 x(k) + E_5 \tag{5c}$$

where $u = (v_c, u_1, u_2)^{\mathrm{T}} \in \mathbb{R} \times \{0, 1\}^2$ is the input vector, and $z(t) \in \mathbb{R}^{22}$ and $\delta(t) \in \{0, 1\}^{10}$ are continuous and binary auxiliary variables, respectively. The auxiliary variables describe the piecewise affine dynamics (3).

3. CONTROL ARCHITECTURE

The architecture of the feedback control system closed over two wireless network links is shown in Figure 4. The solid boxes are the functional blocks while the dashed boxes indicate the implementation on the physical platforms. Temperature data from the physical plant is sent over a WSN to the Host PC. An observer estimates the system states, which are used by the hybrid MPC to compute the control commands. The commands are sent to the Target PC over a WLAN connection. A local controller implemented in the Target PC and in the motor electronics uses the Hybrid MPC control commands to compute the actuation. This section describes the control system in some detail together with the wireless networks and their control and estimation-oriented model.

DOI: 10.1002/rnc

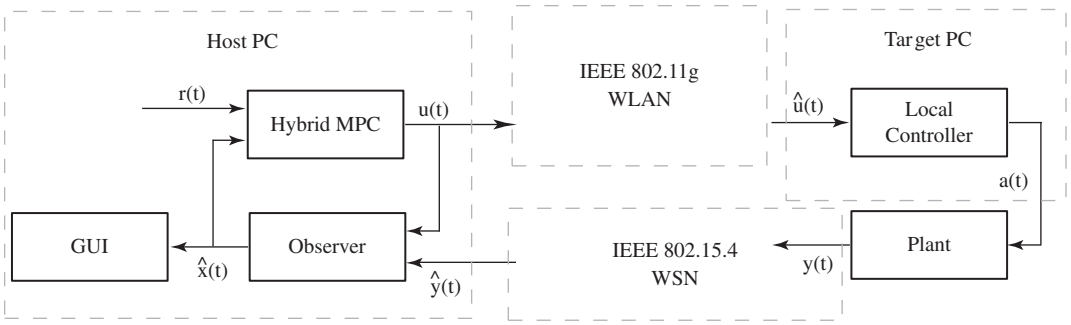

Figure 4. Wireless control architecture.

3.1. Control system

The control architecture is a cascade controller. The inner controller is locally placed at the plant; hence it is referred to as *local controller*. The outer controller is remotely located and referred to as *remote controller*. The remote controller exploits network links to exchange information with the plant and the local controller. The inner (local) controller receives high level commands (setpoints) from the outer (remote) controller, and regulates the process accordingly, ensuring stability and tracking. The outer controller generates the setpoints based on the simplified system model, which results from the plant in closed loop with the local controller. A simplified model makes easier the long-term planning, which is computationally intensive, while stability and tracking need to be enforced by a local controller to avoid delays and data losses. From an implementation point of view the local controller must be fast, but does not need large computational effort. A simple linear controller accomplishes these tasks. On the other hand the remote controller runs on a slower time scale, but needs larger computational effort to generate optimal plans. We implement the remote controller by hybrid MPC. The controller implementation is further described in Section 5.

3.2. Wireless networks

Two networks support the delocalization of the outer controller from the plant site. In this way the remote controller can be implemented in a powerful computer, which does not need to be located in the proximity of the plant. Wireless networks are used to transmit the process measurements from the plant to the remote controller and to transmit the setpoints from the remote controller to the local one. The process measurements are sent over a WSN implemented over the network standard IEEE 802.15.4, while the remote control commands are communicated over IEEE 802.11 g. The radio frequency for both networks is 2.4 GHz, but otherwise there is no interaction between the two networks.

In this paper we use a simple model for the network links based on the *erasure channel*. Let u and y denote the plant input and output, respectively, and $\hat{u}$ and $\hat{y}$ denote the signals after transmission. We have that $\hat{u}(t)=u(t)$ and $\hat{y}(t)=y(t)$, if the corresponding packets at time t are received. We assume that the receivers can detect if a packet is lost (for instance by a suitably calibrated timeout), and that the detection is instantaneous. We denote a lost packet by ε, so that if at time t the command packet was lost, $\hat{u}(t)=\varepsilon$, while if the sensor packet was lost, $\hat{y}(t)=\varepsilon$. In the case the command packet or the sensor packet is lost, the local controller or the estimator, respectively, takes adequate actions to ensure continuing control operations, as described later in Section 4. Note that in this simple network model, we make the reasonable assumption that the communication delays during normal operations are negligible compared with the plant dynamics, while longer delays can be modelled as packet drops, since largely delayed data are not useful for real time control.

Even if the network model we use in this paper is simple, by implementing the stochastic hybrid MPC approach [16] more complex models for the communication links can be used, such as the Gilbert models described in [23]. In this case, provided that

DOI: 10.1002/rnc

an algorithm for network state estimation, such as the ones in [24], is integrated in the control system, the control strategy can compensate for variations in the network characteristics. See [24] for further details on more complex network models, and on network state estimation algorithms.

4. CONTROL SYSTEM DESIGN

The overall control strategy is based on a local controller and on a hybrid MPC algorithm that acts as a reference governor [17, 18]. The local feedback controller is implemented in the proprietary motor electronics. It controls the voltage to track the belt velocity commands received from the remote controller. At the same time, it rejects disturbances due to varying belt friction. Additional control logic is implemented to convert the velocity command received from the remote controller into a command format that can be sent to the servo-controller. The additional logic also ensures continuing operation when the command packet is lost: when the command packet is lost, the local controller applies a zero-order holder, which results in $v_c(t)=v_c(t-1)$.

For the remote controller, we use the Hybrid MPC implemented in the Hybrid Toolbox for MATLAB [25], where the optimization problem is solved online in real time using the mixed-integer quadratic programming solver CPLEX [26]. Note that the laboratory process and the control architecture are general enough for experimentation of other control strategies.

4.1. The hybrid MPC algorithm

In order to be used for prediction in the MPC algorithm, the hybrid discrete-time model developed in Section 2.1 needs to be extended by two additional states. The first one is the ambient temperature T_{amb} in (2). Such a state remains constant in the prediction model and represents a measured disturbance. The second additional state is the 'input memory' state x_u, which is used to constrain the acceleration of the belt, not explicitly modelled in (4). The dynamics of x_u are defined by

$$x_u(t+1)=v_c(t) \tag{6}$$

The acceleration at time t for a given input $v_c(t)$ can be computed by backward Euler approximation from $x_u(t)$ and $v_c(t)$ as $\big(v_c(t)-x_u(t)\big)/T_s$, so that constraints on the acceleration can be expressed as state constraints. The system model becomes

$$x(t+1)=\begin{pmatrix} a_{11} & 0 & 0 & 1 & 0\\ a_{21} & a_{22} & 0 & 1 & 0\\ 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} x(t) + \begin{pmatrix} b_{11}\chi(t)\\ b_{11}\chi(t)\\ b_{32}v_c(t)\\ 0\\ v_c(t) \end{pmatrix} \tag{7a}$$

$$y(t)=\begin{pmatrix} 0 & 1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0 & 0 \end{pmatrix} x(t) \tag{7b}$$

where $x=(T_1, T_2, p, T_{\text{amb}}, x_u)^{\mathrm{T}}$.

The hybrid MPC algorithm [15] is based on the solution at each time step t of the following optimization problem:

$$\begin{aligned} \min\ & J(\{u(k), \delta(k|t), z(k|t)\}_0^{N-1}, x(t)) \\ &\triangleq \sum_{k=0}^{N-1} \left(q_{v_c} v_c(k)^2 + q_z \left(\frac{1}{T_s}\right)^2 (v_c(k)-x_u(k|t))^2 \right. \\ &\quad \left. + \|Q_y(y(k|t)-y_r)\|_2 \right) + q_\rho \rho^2 \end{aligned} \tag{8a}$$

subject to dynamics (5) with state update equation as in (7), and to the state and input constraints

$$\begin{pmatrix} 20\\ 20\\ 0 \end{pmatrix} \leqslant \begin{pmatrix} T_1(k|t)\\ T_2(k|t)\\ p(k|t) \end{pmatrix} \leqslant \begin{pmatrix} 50\\ 50\\ 1.2 \end{pmatrix}, \quad k=1,\ldots,N \tag{8b}$$

$$-0.1 \leqslant v_c(k|t) \leqslant 0.1, \quad k=0,\ldots,N-1 \tag{8c}$$

$$u_1(k|t),\ u_2(k|t) \in \{0,1\}, \quad k=0,\ldots,N-1 \tag{8d}$$

DOI: 10.1002/rnc

where distances are expressed in meters, velocities in meters per second, and temperatures in degrees Celsius. The tuning parameters of the hybrid MPC problem (8d) were selected as

$$N=4, \quad q_\rho=10^3, \quad q_{v_c}=2$$
$$q_z=1, \quad Q_y=\begin{pmatrix}0.01 & 0\\ 0 & 0.6\end{pmatrix}$$

according to the following rationales. The acceleration and the velocity of the belt should be low, in order to reduce power consumption and to avoid wild dynamics, that can cause excessive wear. We want to track a position reference and a temperature reference and we also want the state to remain in a predefined 'safe' set, that excludes high and low temperatures and excessive velocities. The continuous input reference is set to 0 favoring light actuation of the belt. The output reference profile $y_r \in \mathbb{R}^2$ defines the desired behavior of the system. The length of the horizon N affects the performance of the controller. A longer horizon gives a smoother behavior, a shorter one gives a more aggressive controller. Furthermore, the longer the horizon, the more complex the optimization problem, hence the prediction horizon N is chosen by trading off between the performance and the available computational power.

The hybrid MPC algorithm executes the following operations at each time step t:

1. the system output $y(t)$ is measured and the state estimate $\hat{x}(t)$ is computed;
2. the optimal control problem (8) is solved with $x(0|t)=\hat{x}(t)$;
3. the first optimal input $u^*(0)$ is applied to the system as the current control $u(t)$.

4.2. Observer

In the considered process, not all of the states are measurable. Thus, we design an observer to estimate the unmeasured states. The available measurements to use are the belt position, measured using the encoder mounted on the belt, and the temperature registered by the wireless sensors. In the design we use the simplest applicable observer, which is the reduced order nonlinear Luenberger observer

$$\hat{x}(t+1|t+1)=\Phi\hat{x}(t|t)+\xi(t)+K[\hat{y}(t+1) - C(\Phi\hat{x}(t|t)+\xi(t))] \tag{9a}$$

$$\xi(t)=\begin{pmatrix}b_{11}\chi(t|t)\\ b_{21}\chi(t|t)\\ T_s v_c(t)\end{pmatrix}, \quad K=\begin{pmatrix}k_{11} & k_{12}\\ 1 & 0\\ 0 & 1\end{pmatrix} \tag{9b}$$

where $T_s=250\,\text{ms}$ is the sampling period, and $k_{11}=5$, $k_{12}=0$.

As referred in Section 3.2, the wireless network links introduce packet losses. In fact, the observer uses the received signal $\hat{y}$, instead of the locally measured signal y. When packets containing measurements are lost, a simple estimation method is to let the estimation evolve in open loop [27], which means that if no measurements are received, the system is assumed to evolve according to the hybrid prediction model (3)–(4). In this case we set $y=C(\Phi\hat{x}(t|t)+\xi(t))$ in Equation (9), so that (9) becomes equivalent to the open-loop update $\hat{x}(t+1|t+1)=\Phi\hat{x}(t|t)+\xi(t)$. In the proposed control architecture, the packets can also be dropped in the link that connects the controller to the plant, even if, being a TCP/IP wireless link, this is supposed to be more reliable. When packets containing commands are lost, the local controller keeps the previous reference. More advanced strategies for compensating lossy communication are discussed in [28]. How to jointly estimate network states and process states is discussed in [24].

5. IMPLEMENTATION

The closed-loop control system is shown in Figure 4. It consists of five main blocks in addition to two wireless networks. In this section we first describe the hardware and the software architectures. Then, we discuss the controller implementation.

5.1. Hardware architecture

The hardware architecture of the closed-loop system is shown in Figure 5. The remote controller runs in the *Host PC*, which is a 1.2 GHz Pentium-M™ laptop,

DOI: 10.1002/rnc

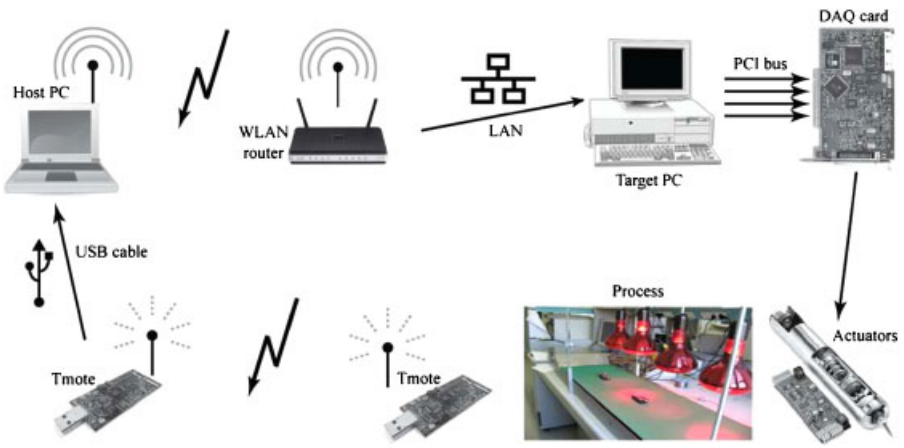

Figure 5. Hardware architecture.

equipped with 632 MB RAM. For communications, it is equipped with an Intel® PRO/Wireless LAN 2100 3A Mini PCI Adapter integrated IEEE 802.11 g WLAN card. The local controller additional logic runs in the *Target PC*, which is a Pentium™ 133 MHz, equipped with 128 MB RAM, only 4 MB of which are actually used because of software architecture limitations. To enable communication with the Host PC, the Target PC is connected via ethernet LAN to a D-Link™ DWL-2100AP WLAN router. To interface the Target PC with the process, a National Instruments® PCI-6024E DAQ-board is used.

The process belt is moved using a belt roller with an encapsulated 24 V DC-motor, which in turn is controlled using a DC servo-amplifier. An incremental angular optical encoder connected to the belt is used to measure the velocity. The switching of the lamps is managed using two relays, one for each pair of lamps, to turn on and off their supply currents. The encoder and all the actuators are connected through the DAQ board.

The objects moving on the belt are Tmote Sky™ wireless sensors from Moteiv® [9]. These, so-called motes are equipped with temperature, humidity, and light sensors, a low-power 8 MHz 16 bit microprocessor, and a 2.4 GHz IEEE 802.15.4 radio transceiver. The mote placed on the belt measures its temperature and communicates it to another mote connected to the USB port of the Host PC. Note that the measurement is sent via a WSN that can be composed of a varying number of motes, depending on how many motes are positioned along the area between the remote controller and the process.

5.2. *Software architecture*

The software architecture of the system is shown in Figure 6. The control application consists of a distributed implementation over four platforms: two of these are implemented on Tmotes and two are implemented on PCs.

The Host PC runs Microsoft® Windows™ XP. On top of this, it runs MATLAB®7.1, CPLEX™9.0, and the Hybrid Toolbox v1.1.0 for executing the MPC algorithm, which solves problem (8). Concurrently to the MPC algorithm, the Host PC runs the Virtual COM software from FTDI Chip® (http://www.ftdichip.com), which reads the USB port of the Host PC Tmote and abstracts a virtual COM RS-232 port. The virtual COM port is in turn read by the Java Sensor Reader program, which reads the port and presents the data in a suitable MATLAB format. We simply denote the software abstraction of the Host PC as *Host*.

The Target PC runs xPC-Target™ real-time kernel [29], with the application being developed in SIMULINK®, and with code generation provided by

DOI: 10.1002/rnc

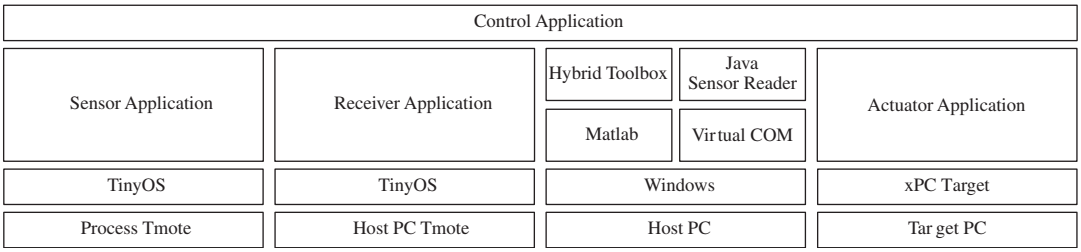

Figure 6. Software architecture.

REAL-TIME WORKSHOP™. The xPC toolbox allows one to use a standard PC, the Target PC in our case, as a microcontroller in a transparent way, which is useful for controller prototyping since it is easy to reconfigure and is inexpensive. It provides a hardware abstraction for TCP/IP communication with the *Host*, as well as an abstraction toward the DAQ card. The full hardware abstraction of the Target PC from the WLAN router to the DAQ is referred to as the *Target*.

Both the Tmotes on the belt and the Tmote connected to the Host PC run TinyOS with custom applications. The Tmotes on the belt run a sensor application software, which samples the onboard temperature sensor at the system sampling frequency 4 Hz and sends the data to the Host PC Tmote. The Host PC Tmote runs a receiver application software, which listens for packets from the sensor application and forwards them to its USB controller connected by the USB port to the *Host*.

5.3. Controller implementation

The Hybrid MPC algorithm is implemented on the *Host* within the Hybrid Toolbox for MATLAB [25]. System model (7) is written in HYSDEL [22] and automatically converted by the associated compiler into the MLD system (5). The optimal control problem (8) is formulated using the Hybrid Toolbox [25] and included into a SIMULINK model as an S-function. The resulting optimization problem consists of 141 optimization variables, 93 continuous and 48 binary, respectively, and 585 mixed-integer linear inequalities. The average time required to solve the optimization problem using CPLEX is 17 ms, with a worst-case computation time of around 125 ms. This computation time motivates the choice of the sampling frequency of 4 Hz for the overall control system. After the control command has been computed, it is sent to the *Target* via the wireless TCP/IP link.

From a functional point of view, the *Target* and the proprietary motor electronics implement the local controller, where the electronics provide the feedback component and the Target computer implements the interfaces and the backup logics needed to counteract command packet losses. The Target generates the motor commands $\hat{u}_c(t)$ from the commanded belt velocity $\hat{v}_c(t)$ received from the Host through the network by performing the inversion $\hat{u}_c(t) = \gamma^{-1}(\hat{v}_c(t))$. The feedback component of the local controller is a servo-controller implemented in the proprietary motor electronics, which regulates the input voltage to the electrical motor to track the desired belt velocity $\hat{v}_c(t)$, using the command $\hat{u}_c(t)$ received from the Target and a local voltage feedback. The local controller rejects disturbances caused by the variable moving part position on the belt and by the variable friction. The Target also integrates the encoder signal to generate the position measurements to be sent to the Host. Finally, the Target implements a zero-order holder on the last command received from the Host. This functionality handles the recovery action for the packet losses that occur in the link from the *Host* to the *Target*, ensuring continuing operation by providing the most recent available command to the process.

6. EXPERIMENTAL RESULTS

In this section we present experimental results of the process in closed loop with the hybrid MPC controller

DOI: 10.1002/rnc

designed in Section 4.1. The experiments aim at evaluating the performance of the control architecture, and, most of all, the impact of the wireless communication on the closed-loop behavior. We first analyze the behavior with respect to data losses in the forward link, i.e. the communication of the input command from the MPC to the local controller, through the WLAN network. Then we analyze the behavior with respect to data losses in the WSN communication, i.e. in the backward (feedback) channel that brings process measurements to the MPC controller.

In the experiments with forward channel losses, we set a constant reference temperature and position reference $y_r = [35, 0.7]^{\mathrm{T}}$. Since the WLAN network used in the process is very reliable, data losses are introduced on purpose by using a real data-loss profile obtained from a sensor network. In this way, even if in the particular network used for the process very few losses occur, we are able to evaluate the loss effects when less reliable networks are used. The hybrid MPC controller (8) is first tuned in simulation also taking into account the packet loss model of Section 3.2. First, the nominal step response (no packet loss, perfect modelling) is simulated. Then, a lossy feedback channel is simulated with the packet loss profile obtained from a real sensor network. The simulation results are shown in Figure 7(a), where the dashed lines indicate the nominal response and the solid line indicates the response of the model with packet losses simulated from real data. The deviation from the nominal behavior that occurs around $t=150$ s is due to a massive packet drop burst. During such an interval, the lamps are commanded off, but the temperature keeps increasing. This means that the packets containing the current commands are being dropped and an older command is being applied. The position is not affected by the drops, because the velocity input is in steady-state conditions (i.e. it is constant) and hence the backup control action is equal to the command that would be received over the network.

The experimental results are shown in Figure 7(b). Other than by the packet losses, the errors are now introduced by external noise and modelling imperfections. In particular, due to the piecewise affine approximation of (2) the input behavior is more aggressive. The reference point y_r, which in the prediction model is achievable despite the input quantization, is not achievable in the real process. As a consequence the controller keeps switching the lamps on and off, and temperature chatters around the equilibrium in a (disturbed) limit cycle. However, the experimental results (solid lines) are still satisfactorily close to the simulation of the nominal model (reported as dashed lines, for comparison).

Next, we consider the case where losses occur in the backward channel, the feedback channel that brings measurements to the MPC controller. In these experiments, some measurements of the sensor network are lost or corrupted and cannot be used by the controller. According to the strategy in Section 4, if the measurement is not received, the process state estimate is updated by prediction only. Hence, when measurements are not received we set $\hat{y}(k+1) = C(\Phi\hat{x}(t|t) + \xi(t))$ in (9), so that the correction term in (9) cancels.

In these experiments, the temperature reference is a square wave with maximum 42°C, minimum 38°C, and frequency 3 mHz. The position reference is a square wave with maximum 0.9 m, minimum 0.5 m and frequency 10 mHz. The initial position is 0 m, the initial temperature is the ambient temperature, and the experiments last 1000 s. Another sensor node is sending data at higher frequency (20 Hz) to disturb the communication of the sensor on the belt. The base station knows the node-ID of the sensor on the belt and it is able to discard the data sent by the other sensor. However, in this process the data sent by the node on the belt are disturbed.

In Figures 8–10, the results for the first experiment are reported. Figure 8 shows the system measurements after packet drops compensation (hence, $\hat{y} = C\hat{x}(k)$ is plotted if $y(k)$ is not received).

The inputs applied to the system are reported in Figure 9, while Figure 10 shows the sensor network communications' performance. In Figure 10(a), the temperature measurements received from the sensor are shown, where a value of -1 indicates that the temperature measurement is not received. In Figure 10(b), the packet reception rate (PRR) as a function of time is reported. $\mathrm{PRR}(t)$ is computed as the ratio between the number of received packets and the number of sent packets during the time interval $[t-15\,\mathrm{s}, t+15\,\mathrm{s}]$ (for this reason $\mathrm{PRR}(t)$ is shown for $t \geqslant 15$ s). In this first experiment, the network is relatively reliable as shown

DOI: 10.1002/rnc

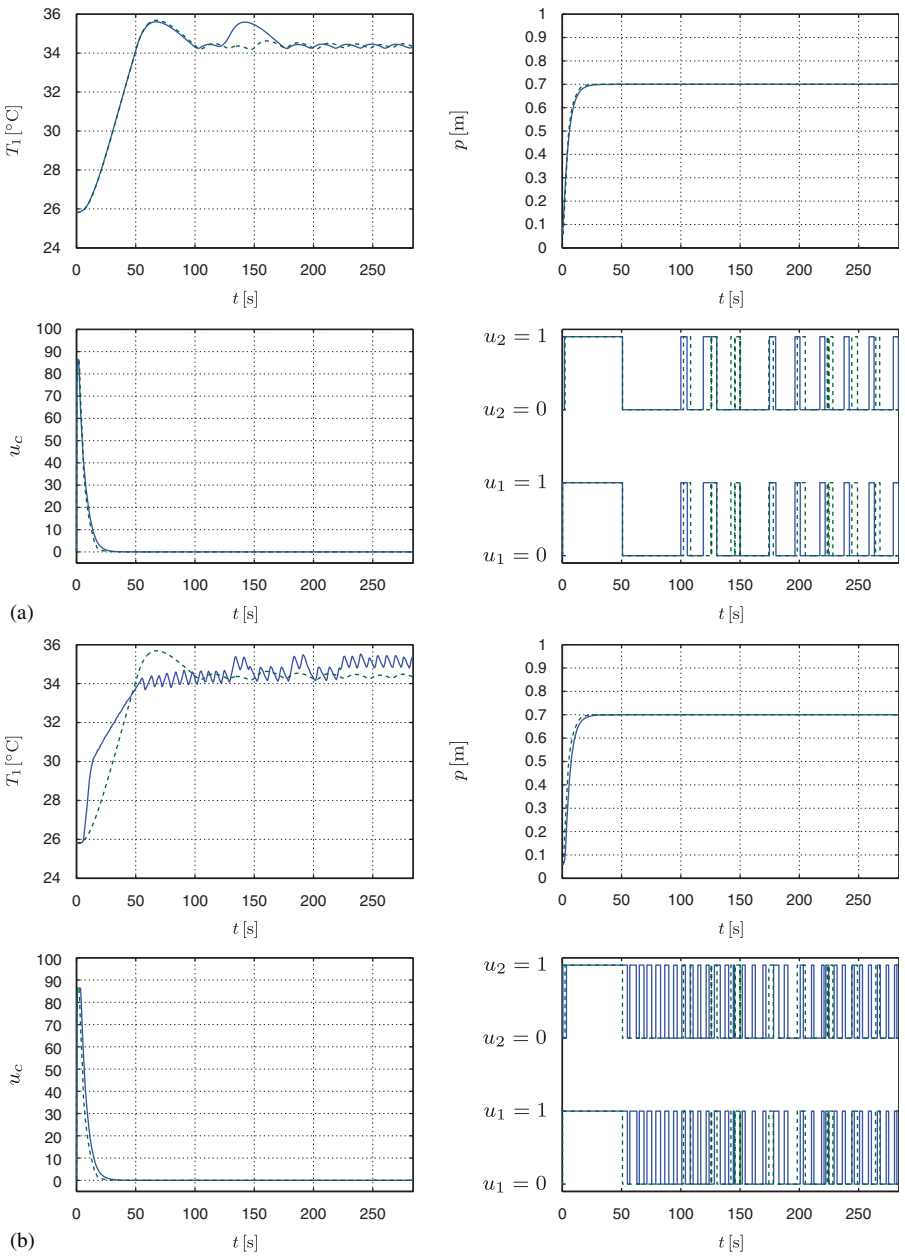

Figure 7. Simulations and experiments on the process shown in Figure 2. (a) Simulations. Nominal behavior (dashed) and behavior with feedback packet losses simulated from real packet loss profile (solid). (b) Experimental behavior (solid) and nominal simulated behavior (dashed).

DOI: 10.1002/rnc

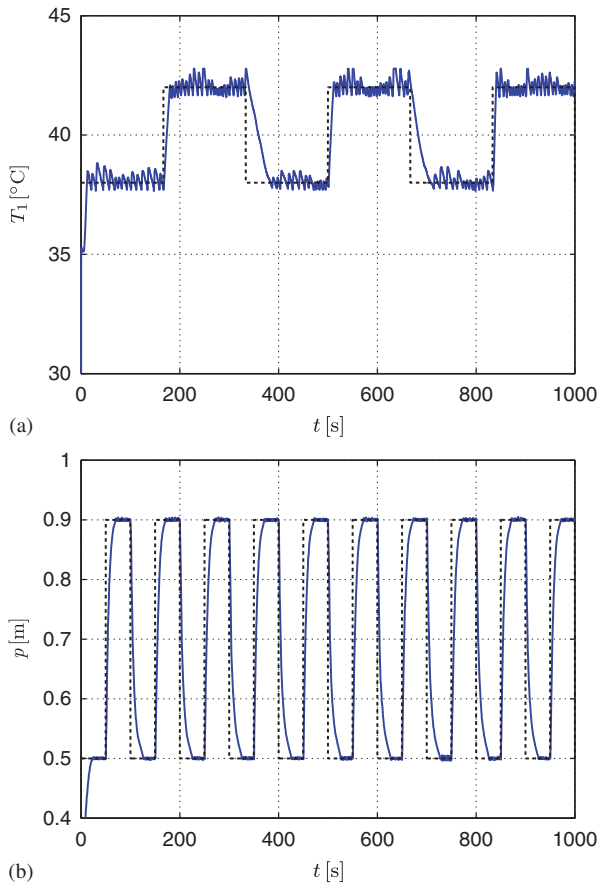

Figure 8. Experiments with losses in the feedback channel, reliable channel case: temperature and position (solid) and corresponding references (dashed) as available for the remote controller. (a) Temperature: in case the measurement is not received, the predicted value is shown. (b) Position: no losses occur in the WLAN network that carries position measurements.

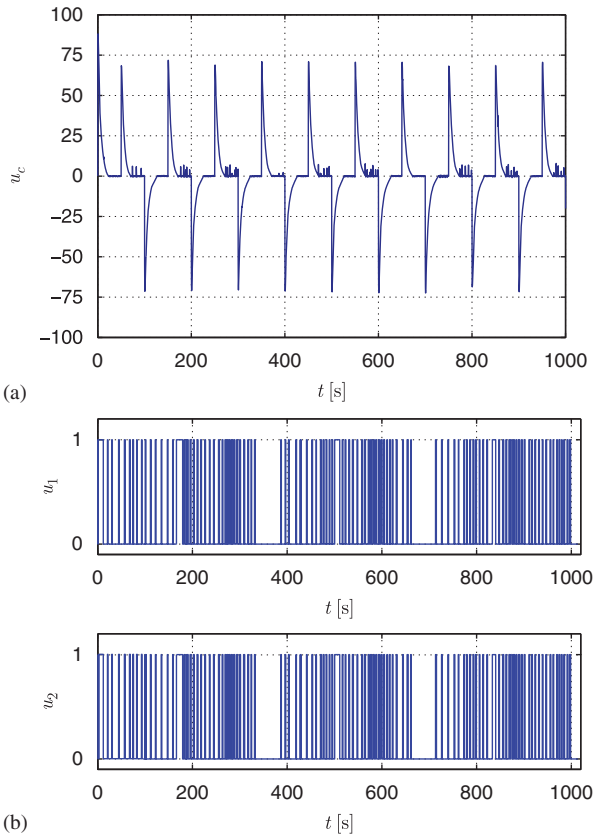

Figure 9. Experiments with losses in the feedback channel, reliable channel case. commands issued by MPC: (a) belt motor command, and (b) lamp commands.

by Figure 10(b), and in fact only about 7.9% of the measurements collected along the whole experiment are lost. The reference tracking performance is good and the oscillations around the setpoint are due to the quantization of the control input that affects the temperature (the lamps that are switched on/off). The tracking performance of the position is even higher, because the position measurement is sent through the WLAN network, which is very reliable. In fact, no data loss occurs for the position measurements during this experiment, and the delays are in the order of milliseconds, hence much smaller than the sampling period.

In Figures 11–13, the results for the second experiment are reported. Figure 11 shows the process measurements after packet drops compensation, similar to Figure 8. Figure 12 reports the inputs applied to the system, and Figure 13 shows the sensor network communications' performance. In Figure 13(a), the temperature measurements received from the sensor are shown, similar to Figure 10(a). In Figure 13(b), the PRR computed during a moving centered window of 30 s is reported, similar to Figure 10(b). In the second experiment, the feedback network link is made

DOI: 10.1002/rnc

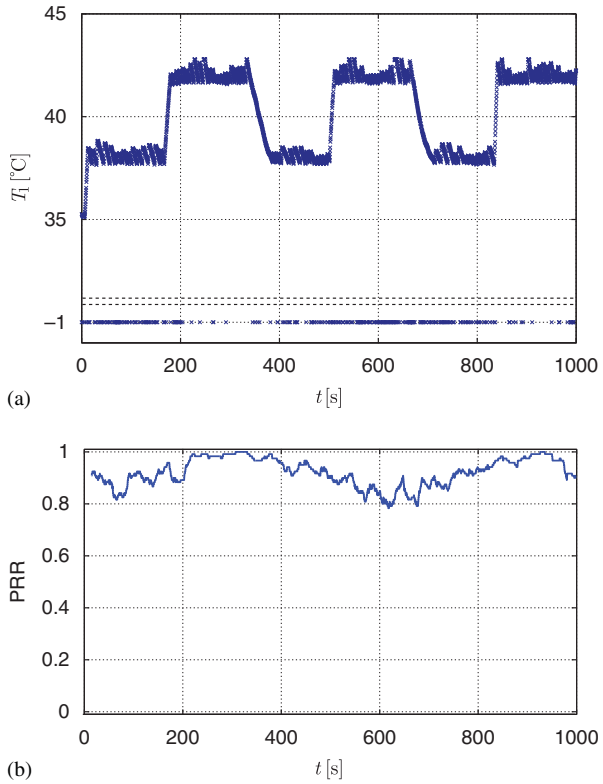

Figure 10. Experiments with losses in the feedback channel, reliable channel case: measurements received from the network. (a) Measurements received by the controller. A reported value -1 indicates the measurement is not received. (b) Packet reception rate PRR(t) computed over a moving centered window of 30 s.

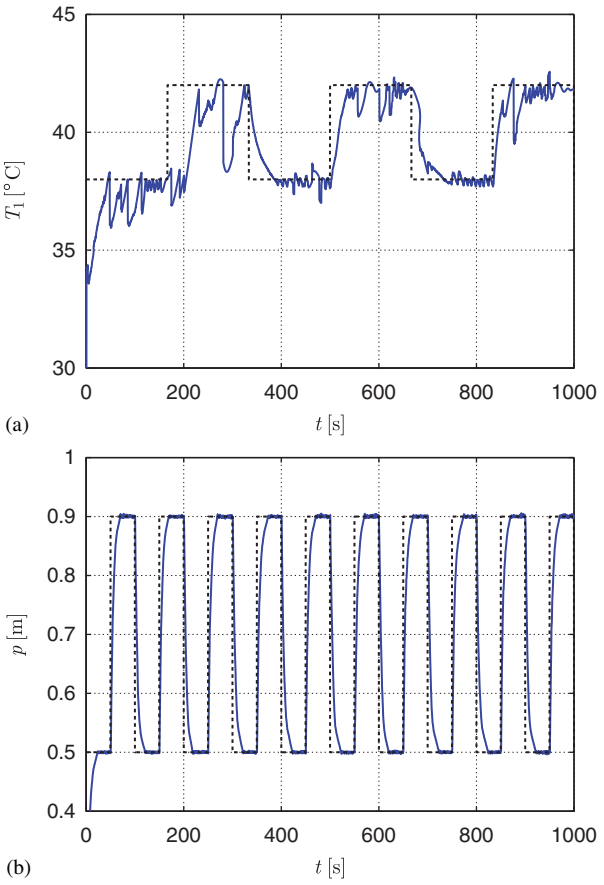

Figure 11. Experiments with losses in the feedback channel, unreliable channel case: temperature and position (solid) and corresponding references (dashed) as available for the remote controller. (a) Temperature: in case the measurement is not received, the predicted value is shown. (b) Position: no losses occur in the WLAN network that carries position measurements.

unreliable by means of reflective aluminum foils around the sensor antenna that disrupt the signals. The PRR is much lower in Figure 13(b) than in Figure 10(b). The total percentage of missing temperature measurements is now 62.8%. This affects the temperature reference tracking performance as it appears from Figure 11(a). In particular, the abrupt changes in the temperature value in Figure 11(a) reveal long bursts of missing data. When data are missing, the predicted value of the temperature is shown in Figure 11(a). As a consequence, when a measurement is finally received, the real temperature is shown, which results in a jump in the plot. A closer look at one of these events is shown in Figure 14, which reports the temperature used as available for the controller (solid line), the temperature reference (dashed line), and the received temperature sensor data (crosses) during one of the intervals where the PRR is lower (cf. Figure 13(b)). Between $t=550$ s and $t=600$ s, very few data are received. The prediction model seems to be sufficiently good when the temperature has to be increased (e.g. between $t=560$ s and $t=580$ s), but when the temperature has to remain constant or has to be decreased (e.g. between $t=580$ s and $t=600$ s) there is a large difference between the

DOI: 10.1002/rnc

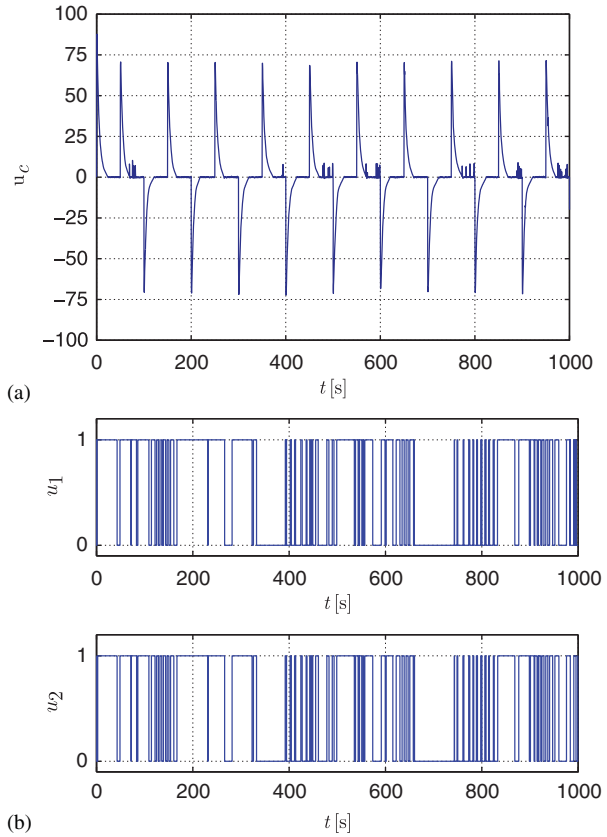

Figure 12. Experiments with losses in the feedback channel, unreliable channel case. commands issued by MPC: (a) belt motor command, and (b) lamp commands.

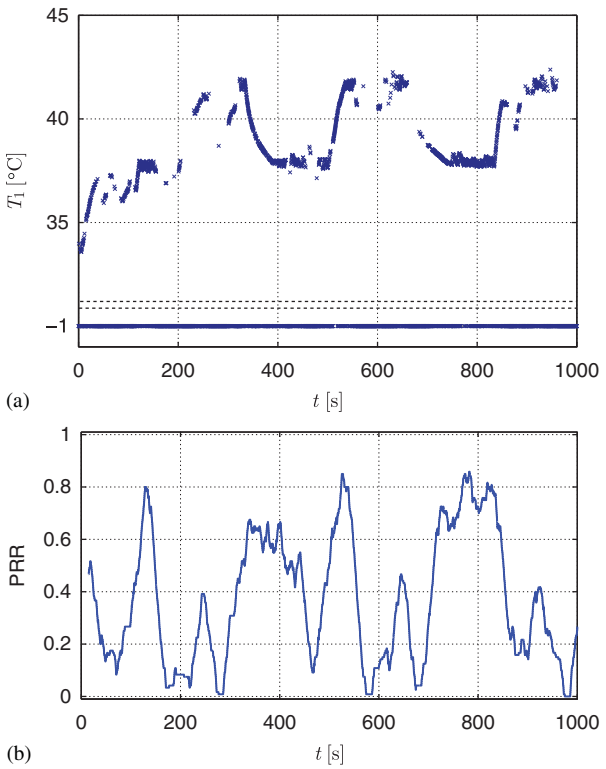

Figure 13. Experiments with losses in the feedback channel, unreliable channel case: measurements received from the network. (a) Temperature measurements received by the controller. A reported value −1 indicates that the measurement is not received. (b) Packet reception rate PRR(t) computed over a moving, centered window of 30 s.

prediction model and the real process, which results in the jump at $t = 600$ s, when the first measurement is received after more than 20 s of data losses.

7. CONCLUSIONS AND FUTURE RESEARCH

This paper has presented a hybrid MPC design and an experimental demonstration of remote control over wireless networks. Data packets dropped in both forward and feedback communication links can be handled with good results using standard hybrid MPC techniques. The hybrid MPC design has several advantages compared with traditional controllers. The most obvious advantage is that it offers the possibility to handle process nonlinearities and on/off inputs, and to enforce constraints on states, inputs, and outputs in a simple and direct way. The setup has been proven easy to tune. The only drawback is that hybrid MPC can be computationally intense, although it is fast enough for the application at hand, and the worst-case computation period can be bounded *a priori* by imposing time constraints on the optimization solver. The computational burden required by hybrid MPC is one of the main motivations for investigating its network-based implementation, since this allows to execute the MPC algorithms in powerful computational units placed remotely with respect to the plant, and connected to a

DOI: 10.1002/rnc

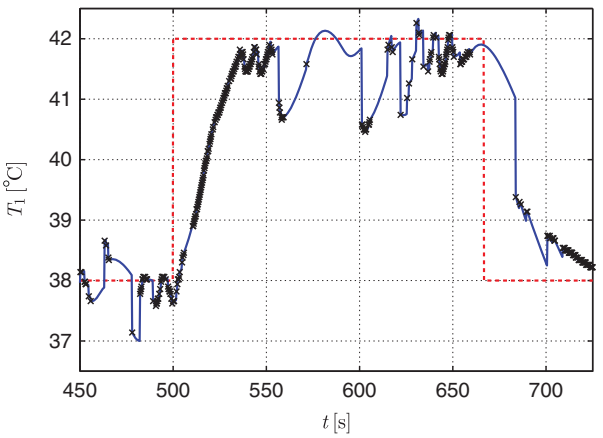

Figure 14. Experiments with losses in the feedback channel, unreliable channel case: a closer view to temperature reference tracking performance as affected by dropped data.

much simpler local controller at the plant by flexible and inexpensive (wireless) network links.

Although the proposed control architecture ensures continuing operation despite data losses in both the forward and the feedback channel, long sequences of packet drops may degrade the overall system performance. This can be seen in the experimental results. For this reason, the authors are currently working on the implementation of the stochastic hybrid MPC controller [16] where the statistical properties of the communication channel will be estimated as proposed in [24], and used to adaptively constrain the working set of the controller to ensure robustness with respect to network condition variability.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge Francesco Molendi for his assistance in programming the motes, Davide Barcelli, and Luca Lucherini for building most of the laboratory experiment, and Pan Gun Park for providing the wireless sensor network packet drop sequences used for simulating forward channel packet drops. S. Di Cairano and A. Bemporad were supported by 'Advanced control methodologies for hybrid dynamical systems' PRIN'2005, of the Italian Ministry for University and Research (MIUR). K. H. Johansson, and E. Henriksson were supported by SOCRADES Integrated Project of the European Commission, the Swedish Research Council, the Swedish Governmental Agency for Innovation Systems, and the Swedish Foundation for Strategic Research.

REFERENCES

1. Samad T, McLaughlin P, Lu J. System architecture for process automation: review and trends. *Journal of Process Control* 2007; **17**:191–201.
2. Kumar PR. New technological vistas for systems and control: the example of wireless networks. *IEEE Control Systems Magazine* 2001; **21**:24–37.
3. Årzén KE, Bicchi A, Dini G, Hailes S, Johansson KH, Lygeros J, Tzes A. A component-based approach to the design of networked control systems. *European Journal of Control* 2007; **13**(2–3):261–279.
4. WirelessHART. Available from: http://www.hartcomm2.org.
5. ISA-SP10014 Wireless Networks Optimized for Industrial Monitoring. Available from: http://www.isa.org/filestore/ISASP100_14_CFP_14Jul06_ Final(2).pdf.
6. SOCRADES. Integrated Project. EU 6th Framework Programme. Available from: http://www.socrades.eu/.
7. Johansson KH. Closing the loop over wireless networks: fundamentals and applications. *International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks* (*WiOpt*), Berlin, Germany, 2008. Keynote presentation.
8. Morse J. Wireless in industrial systems: cautious enthusiasm. *Industrial Embedded Systems* 2006; **2**(2):10–11.
9. Moteiv Corporation. Tmote sky March 2007. Available from: http://www.moteiv.com/products/tmotesky.php.
10. Sinopoli B, Sharp C, Schenato L, Schaffert S, Sastry SS. Distributed control applications within sensor networks. *Proceedings of the IEEE* 2003; **91**:1235–1246.
11. Liu X, Goldsmith A. Wireless network design for distributed control. *Proceedings of the 43th IEEE Conference on Decision and Control*, Paradise Island, Bahamas, 2004; 2823–2829.
12. Kawadia V, Kumar PR. A cautionary perspective on cross layer design. *IEEE Wireless Communication Magazine* 2005; **12**:3–11.
13. Rawlings JB. Tutorial overview of model predictive control. *IEEE Control Systems Magazine* 2000; **20**(3):38–52.
14. Qin SJ, Badgwell T. A survey of industrial model predictive control technology. *Control Engineering Practice* 2003; **93**(316):733–764.
15. Bemporad A, Morari M. Control of systems integrating logic, dynamics, and constraints. *Automatica* 1999; **35**(3):407–427.
16. Bemporad A, Di Cairano S. Optimal control of discrete hybrid stochastic automata. In *Hybrid Systems*: *Computation and Control*, Morari M, Thiele L (eds). Lecture Notes in Computer Science, vol. 3414. Springer: Berlin, 2005; 151–167.
17. Bemporad A, Casavola A, Mosca E. Nonlinear control of constrained linear systems via predictive reference management. *IEEE Transactions on Automatic Control* 1997; **AC-42**(3):340–349.

DOI: 10.1002/rnc

18. Gilbert EG, Kolmanovsky IV. Fast reference governors for systems with state and control constraints and disturbance inputs. *International Journal of Robust Nonlinear Control* 1999; **9**(15):1117–1141.
19. Bemporad A. Predictive control of teleoperated constrained systems with unbounded communication delays. *Proceedings of the 37th IEEE Conference on Decision and Control*, Tampa, FL, 1998; 2133–2138.
20. Danesi A, Fagiolini A, Savino I, Pallottino L, Schiavi R, Dini G, Bicchi A. A scalable platform for safe and secure decentralized traffic management of multiagent mobile systems. *ACM Workshop on Real-world Wireless Sensor Networks*, Upsaala, Sweden, 2006.
21. Bemporad A, Di Cairano S, Henriksson E, Johansson KH. Hybrid model predictive control based on wireless sensor feedback: an experimental study. *Proceedings of the 46th IEEE Conference on Decision and Control*, New Orleans, LA, 2007; 5062–5067.
22. Torrisi FD, Bemporad A. HYSDEL—a tool for generating computational hybrid models. *IEEE Transactions on Control Systems Technology* 2004; **12**(2):235–249.
23. Yu X, Modestino J, Tian X. The accuracy of Gilbert models in predicting packet-loss statistics for a single-multiplexer network model. *INFOCOM—24th Conference of IEEE Computer and Communications Societies*, Miami, FL, 2005; 2602–2612.
24. Di Cairano S, Johansson KH, Bemporad A, Murray R. Dynamic network state estimation in networked control systems. In *Hybrid Systems*: *Computation and Control*, Egerstedt M, Mishra B (eds). Lecture Notes in Computer Science, vol. 4981. Springer: Berlin, Heidelberg, 2008; 144–157.
25. Bemporad A. *Hybrid Toolbox—User's Guide*. 2003. Available from: http://www.dii.unisi.it/hybrid/toolbox.
26. ILOG. Inc. *CPLsEX 9.0 User Manual*. Gentilly Cedex, France, 2004.
27. Hespanha JP, Naghshtabrizi P, Xu Y. A survey of recent results in networked control systems. *Proceedings of IEEE Special Issue on Technology of Networked Control Systems* 2007; **95**(1):138–162. ISSN: 0018-9219.
28. Henriksson E, Sandberg H, Johansson KH. Predictive compensation for communication outages in networked control systems. *Proceedings of the 47th IEEE Conference on Decision and Control*, Cancun, Mexico, 2008; 2063–2068.
29. The Mathworks Inc. *xPC Target—For Use with Real-time Workshop* 2000. User's Guide—Version 1.1.

DOI: 10.1002/rnc