A packet-based dual-rate PID control strategy for a slow-rate sensing Networked Control System.

Abstract

This paper introduces a packet-based dual-rate control strategy to face time-varying network-induced delays, packet dropouts and packet disorder in a Networked Control System. Slow-rate sensing enables to achieve energy saving and to avoid packet disorder. Fast-rate actuation makes reaching the desired control performance possible. The dual-rate PID controller is split into two parts: a slow-rate PI controller located at the remote side (with no permanent communication to the plant) and a fast-rate PD controller located at the local side. The remote side also includes a prediction stage in order to generate the packet of future, estimated slow-rate control actions. These actions are sent to the local side and converted to fast-rate ones to be used when a packet does not arrive at this side due to the network-induced delay or due to occurring dropouts. The proposed control solution is able to approximately reach the nominal (no-delay, no-dropout) performance despite the existence of time-varying delays and packet dropouts. Control system stability is ensured in terms of probabilistic Linear Matrix Inequalities (LMIs). Via real-time control for a Cartesian robot, results clearly reveal the superiority of the control solution compared to a previous proposal by authors.

Keywords

Networked control systems; slow sensors; non-uniform sampling; multi-rate control; packet-based control.

1. - Introduction

One of the main features of a Networked Control System (NCS) [1,2,3] is to send data via a shared communication medium in order to control a process. In the last few years, interesting applications have been reported in this field [4,5,6,7]. The present work considers a low cost distributed sensor net, which sends data to a powerful remote server (with high computation and information storage features). One of the main goals of the contribution is to achieve energy saving in this kind of set-up. Energy saving is a crucial issue, especially in battery-powered wireless sensors, having positive effects on their battery lives. As it is well known, the energy consumption in sensor networks is usually due to sensing, processing and communication operations. The last ones consume most part of this amount of energy. Therefore, in order to save energy, a good option is to reduce the data flow frequency through the net. However, a quite low transmission frequency could imply failure to fulfil control specifications, or even process instability. Different control proposals have been introduced to reduce the communication rate preserving the control performance. One of them is the packet-based approach [8,9,10,11], which enables a sequence of signals to be sent over the network simultaneously. In the same way, techniques based on sending data when some thresholds are exceeded [12,13,14] –i.e. event-based control-, or just considering some specific data priority [15], have been used. Another possible option is the so called “multi-rate control”, which is a control technique able to assume different rates for different control loop signals [16,17]. The present work combines multi-rate and packet-based control techniques for dealing, in addition, with some typical NCS problems such as time-varying network-induced delays, packet dropouts, and packet disorder.

The considered NCS locates the low cost sensor net, the process and the actuator (with a low computation power) in the local side, whereas the server (with high computation power and other capabilities) is situated in the remote side. Sensed values must travel through the main network from the local side to the remote side, and control signals, from the remote side to the local side. As it can be said, to reduce energy, a slow transmission frequency was assumed in both links.

Therefore, slow-rate sensed values and slow-rate control actions travel through the NCS. Adopting a dual-rate control strategy, an N faster control updating (being N a positive integer greater than one) can be assumed at the actuator device by converting the slow-rate control signal into a fast-rate one, which enables to achieve the desired control performance. This proposal requires a special controller, actually a non-conventional controller, which is able to compute control actions at period T from signals taken at period NT [17]. In this work, due to the wide knowledge of PID controllers in industrial and academic environments, a non-conventional PID control structure is taken into account. The controller is split into two parts: a slow-rate PI controller and a fast-rate PD controller. The integral action is applied at slow rate because it usually operates at this frequency zone, and hence, it is located at the remote (server) side. The derivative actions, which are associated with faster dynamics, are applied at fast rate, and hence, the PD controller is located at the local side (inside the actuator). The basic design procedure can be looked up in [17,18,19].

The additional difficulties caused by the shared communication medium (network-induced delays, packet dropouts, and packet disorder) can appear depending on the network protocol used in a specific application. Regarding the delays, these are due to waiting-transmission-picking up times and are time-varying during the course of the application. As it is well-known, not compensating for the delays can imply a worsening of the control performance. This problem can be overcome by using, for example, gain-scheduling control strategies such as the one used in [20]. This proposal allows obtaining a simple delay-dependent control law to retune the parameters of the fast-rate PD controller from the round-trip time delay. As the actuator is assumed to include low computation capabilities, it is able to measure the round-trip time delay and compute the fast-rate control signal. In this way, the nominal (no-delay) control performance can be closely maintained despite actuating in a non-uniform way due to the delays (that is, the last control action computed in the previous sensor period is held until new, current control actions are applied after the delay).

Concerning the packet disorder, once again, the fact of choosing a slow sensing period NT (concretely, the sensing period will be longer than the longest round-trip time delay) is beneficial, since it can avoid this phenomenon. Obviously, the decision needs to perform some off-line experiences with usual operation conditions to detect some delay features. Sometimes, a statistical distribution of the network-induced delay is even found [21,22]. As previously commented, despite the fact of choosing the slow sensing period in this way, dual-rate control techniques (with faster control period) enable to achieve a satisfactory performance [17,18]. And, once the sensing period and the rest of NCS parameters are defined, LMIs can be used to ensure control system stability.

Finally, with regard to packet dropouts, if a network protocol like UDP is assumed, this phenomenon can occur [21,22]. In this work, a packet dropout can be derived either from an effective loss or from the expiration of a maximum waiting time. In addition, an upper-bound for consecutive packet dropouts M can be established from the off-line experiences. In order to deal with up to M possible packet losses, a prediction stage is included at the remote side, which contains a state resetting procedure [23,24] in order to face even unstable processes (or marginally stable ones, such as the position output for DC motors governing the axis of a Cartesian robot, which is used in this work). If a packet dropout occurs, a different solution is implemented for each network link. For the local-to-remote link, the process estimator provides the estimated output value in order to generate the current (estimated) PI control action. For the remote-to-local link, a packet-based strategy is adopted. The packet includes the current PI control action and M future ones to be used just in case the dropout occurs.

Summarizing, the main motivation of this paper is to introduce some control techniques (multi-rate and packet-based ones) in order to save energy (reducing the number of transmissions) while maintaining a satisfactory control performance in a NCS, where some typical problems (time-varying network-induced delays, packet dropouts, and packet disorder) must be faced.

The main contributions of this work with regard to another proposal in [20] are: 1) the remote-to-local network link is now considered, which allows not only to distribute the controller between remote and local sides, but also to deal with remote-to-local delays; 2) the network-induced delay is now stochastic. Then, probabilistic LMIs are required to ensure control system stability; 3) packet dropouts are considered in both network links. They are faced via a prediction technique which contains a state resetting procedure in order to even cope with unstable processes.

The paper is structured in the following sections. In section 2, the problem scenario is formally introduced. In section 3, control techniques used in the remote and local sides are presented. Control system stability is enunciated in terms of probabilistic LMIs in section 4. Simulation results in section 5 illustrate the benefits of the proposed control strategy. Section 6 validates the results using a real physical process. Finally, conclusions close this contribution.

NOMENCLATURE	
N, *t*, *T*, *NT*	Multiplicity, and basic, actuation and sensor periods
$\tau_k^{lr}, \tau_k^{rl}, \tau_k$	Local-to-remote, remote-to-local, and round-trip time network-induced delay
d_k^{lr}, d_k^{rl}	Local-to-remote and remote-to-local packet dropout occurrence
$y_k^{NT}, \hat{y}_k^{NT}, \dot{y}_k^{NT}$	Actual, estimated, and actual or estimated, system output (at *NT*)
r_k^{NT}	Reference signal (at *NT*)
$e_{PI,k}^{NT}$	Error signal (PI input) (at *NT*)
$x_k^{NT}, \hat{x}_k^{NT}, \dot{x}_k^{NT}$	Actual, estimated, and actual or estimated, system state (at *NT*)
$u_{PI,k}^{NT}, \hat{u}_{PI,k}^{NT}, \dot{u}_{PI,k}^{NT}$	Actual, estimated, and actual or estimated, PI control action (at *NT*)
$\tilde{u}_{PI,k}^{T}$	Expanded PI control action (at *T*)
$\bar{u}_{PI,k}^{T}, \hat{\bar{u}}_{PI,k}^{T}, \dot{\bar{u}}_{PI,k}^{T}$	Actual, estimated, and actual or estimated, PI action converted at *T*

$u_{PD,k}^{T,\tau_k}, \hat{u}_{PD,k}^{T,\tau_k}, \dot{u}_{PD,k}^{T,\tau_k}$	Actual, estimated, and actual or estimated, (retuned) PD control action (at *T*)
M	Upper bound for estimations
z	*t*-unit operator
$\overline{z} = z^L, \quad L = \frac{T}{t}$	*T*-unit operator
$z_N = \overline{z}^N = z^{N \cdot L}$	*NT*-unit operator

2. - Problem description

The proposed NCS is depicted in Figure 1, where the network is placed between the remote and local sides, and can introduce time-varying delays, packet dropouts and packet disorder. The round-trip time delay for the packet sampled at the instant *kNT* (where $k \in \mathbb{N}$, *T* is the actuation period, *NT* is the sensor period, and $N \in \mathbb{Z}^+$ is a parameter known as multiplicity in a dual-rate control framework [17]) is defined as

$$\tau_k = \tau_k^{lr} + \tau_k^{rl} + \tau_k^c, \tag{2.1}$$

where τ_k^c can be considered as a negligible computation time delay, τ_k^{lr} is the local-to-remote network-induced delay, and τ_k^{rl}, the remote-to-local one. To avoid packet disorder, $\tau_k \in [0, \tau_{\max}]$ must fulfill $\tau_{\max} < NT$. Since in this work, an IP network which uses UDP as the transport layer protocol is taken into account, the distribution of the round-trip time delay is a constant plus a Gamma distributed random variable, whose shape and scale parameters change with load and network segment [22]. Usually, this distribution is approximated as a generalized exponential distribution [25], whose probability density function can take this form

$$P[\tau_k] = \begin{cases} \frac{1}{\phi} e^{\frac{-(\tau_k - \eta)}{\phi}} & , \tau_k \geq \eta \\ 0 & , \tau_k < \eta \end{cases} , \tag{2.2}$$

being the expected value of the delay $E[\tau_k] = \phi + \eta$, and its variance $V[\tau_k] = \phi^2$. A feasible choice of η is the median of the delay. From η and an experimental value of $E[\tau_k]$ (or the mean), ϕ can be easily approximated. Note that a common timer is supposed to be shared by the local devices in such a way that all of them are perfectly synchronized. Then, τ_k can be measured subtracting packet sending and receiving times, not requiring time-stamping techniques.

As well-known, when using the UDP transmission model, packet dropouts appear. This phenomenon is essentially random [21], and hence, it can be modeled as a Bernoulli distribution [1]. The variable d_k^{lr} indicates the possible loss of the packet sent from the local side to the remote one at the instant *kNT* (similarly, d_k^{rl} is defined for the opposite network link). In this work, both variables are considered as a Bernoulli process with probability of dropout:

$$\begin{aligned} p^{lr} &= \Pr[d_k^{lr} = 0] \in [0,1) \\ p^{rl} &= \Pr[d_k^{rl} = 0] \in [0,1) \end{aligned} , \tag{2.3}$$

Next, the different devices included in Figure 1 for the NCS are presented:

- a process to be controlled: as it will be introduced in section 5 and 6, a Cartesian robot will be used.
- a sensor, working at period *NT*, to sample the process output y_k^{NT}. Sensing at this slow rate enables to achieve energy saving by reducing network load.
- a slow-rate PI controller, which generates a PI control action $u_{PI,k}^{NT}$ from the reference r_k^{NT} and the sample y_k^{NT}, as long as it arrives to the remote side ($d_k^{lr} = 1$) after τ_k^{lr}. Otherwise ($d_k^{lr} = 0$), a previously estimated PI control action $\hat{u}_{PI,k}^{NT}$ will be used. Note that to detect a packet dropout in this device, a maximum waiting time $\tau_{\max}^{lr}$ is considered. If $\tau_{\max}^{lr}$ expires and the packet does not arrive, it will be considered as a dropout. More information about

the definition and operation mode of the slow-rate PI controller can be found in subsection 3.1.

- a prediction stage, which computes an array of M estimated, future PI control actions $\left[\hat{u}^{NT}_{PI,k+1},\hat{u}^{NT}_{PI,k+2},\ldots\hat{u}^{NT}_{PI,k+M}\right]$ from the array of the actual and future references $\left[r^{NT}_{k},r^{NT}_{k+1},r^{NT}_{k+2,},\ldots r^{NT}_{k+M}\right]$, and the actual (or estimated) PI control action $u^{NT}_{PI,k}$ (or $\hat{u}^{NT}_{PI,k}$), the process output y^{NT}_{k} (or $\hat{y}^{NT}_{k}$), and the state x^{NT}_{k} (or $\hat{x}^{NT}_{k}$). For the sake of simplicity and brevity, both cases (actual and estimated) will be contained under the notation $\dot{u}^{NT}_{PI,k},\dot{y}^{NT}_{k},\dot{x}^{NT}_{k}$ on the sequel. The main goal of the prediction stage is facing packet dropouts for both network links. A more detailed scheme about this stage is depicted in Figure 2, where a prediction cascade structure is considered. For more information about how the prediction stage works, see subsection 3.4.
- a packet generator, which enables to consider a packet-based control strategy by creating the packet to be sent to the local side, containing $\left[\dot{u}^{NT}_{PI,k},\hat{u}^{NT}_{PI,k+1},\hat{u}^{NT}_{PI,k+2},\ldots\hat{u}^{NT}_{PI,k+M}\right]$.
- a non-uniform actuator which includes a rate converter and a fast-rate PD controller: the rate converter converts the slow-rate PI control signal to a fast-rate one in order to be used as an input to the fast-rate PD controller (more details in subsection 3.2). The two main goals of the fast-rate PD controller are: 1) to achieve the required performance from a slow-rate sensing but acting at a fast rate (that is, a dual-rate control strategy [17]), 2) to compensate for the round-trip time delay, which can be measured at the local side. Depending on d^{rl}_{k}, two different cases can be considered. Figure 3 illustrates the case when no packet dropout occurs from remote to local sides ($d^{rl}_{k}=1$), and Figure 4, the opposite case ($d^{rl}_{k}=0$). In both cases, once the delay τ_k is known (or the packet dropout is detected after expiring a supposed maximum waiting time $\tau_{\max}$, and hence $\tau_k=\tau_{\max}$), the controller generates the PD control signal ($\left[\dot{u}^{T,\tau_k}_{PD,k},\dot{u}^{T,\tau_k}_{PD,k+1},\ldots,\dot{u}^{T,\tau_k}_{PD,k+N-1}\right]$ or $\left[\hat{u}^{T,\tau_k}_{PD,k},\hat{u}^{T,\tau_k}_{PD,k+1},\ldots,\hat{u}^{T,\tau_k}_{PD,k+N-1}\right]$,

depending on d_k^{rl}) by adopting a gain scheduling approach [20]. The control signal is applied to the process by the actuator in a non-uniform way, since the last control action of the previous sensor period *NT*, $\dot{u}_{PD,k-1}^{T,\tau_k}$, is held while no new one can be applied due to the delay τ_k. Note, $\tau_{\max}$ could fulfill $\tau_{\max} \geq dT, d \in \mathbb{Z}^+$, then the first *d* control actions would not be applied, and hence, the control performance could worsen. More information about the fast-rate PD controller can be found in subsection 3.3.

3. - Packet-based control strategy. Prediction stage.

In this section, the packet-based control strategy proposed in this work is formulated in subsections 3.1 (slow-rate PI controller), 3.2 (rate converter), and 3.3 (fast-rate PD controller). Since packet dropouts could occur in both network links, each control stage must consider two cases: a) no packet dropout, b) packet dropout. At the end of the section (in subsection 3.4), the prediction stage is enunciated.

First of all, let us define the transfer function of the continuous plant to be controlled as $G_p(s)$. By using the Z-transform at different periods plus a zero order hold device $H(s)$, different discrete-time versions for *G*$G_p(s)$ can be calculated. Therefore, denoting $\bar{z} = z^L$ and $z_N = \bar{z}^N = z^{N \cdot L}$:

$$G^{NT}(z_N) \triangleq Z_{NT}\left[H_{NT}G_p(s)\right] = \frac{Y^{NT}\left(z_N\right)}{U^{NT}\left(z_N\right)} ;\; G^{T}(\bar{z}) \triangleq Z_{T}\left[H_{T}G_p(s)\right] = \frac{Y^{T}\left(\bar{z}\right)}{U^{T}\left(\bar{z}\right)}$$

$$G^{t}(z) \triangleq Z_{t}\left[H_{t}G_p(s)\right] = \frac{Y^{t}\left(z\right)}{U^{t}\left(z\right)} ,\quad t < T:\quad t \cdot L = T, \quad L \in \mathbb{Z}^+ , \tag{3.1}$$

In addition, the consequent state-space representations for each case can be enunciated as

$$\begin{cases} x_{k+1}^{NT} = A^{NT} x_k^{NT} + B^{NT} u_k^{NT} \\ y_k^{NT} = C^{NT} x_k^{NT} \end{cases} ; \begin{cases} x_{k+1}^{T} = A^{T} x_k^{T} + B^{T} u_k^{T} \\ y_k^{T} = C^{T} x_k^{T} \end{cases} ; \begin{cases} x_{k+1}^{t} = A^{t} x_k^{t} + B^{t} u_k^{t} \\ y_k^{t} = C^{t} x_k^{t} \end{cases} , \tag{3.2a}$$

Secondly, let us consider a continuous PID controller which is designed according to classical methods in order to achieve certain specifications for the process to be controlled. This is the configuration considered for the continuous PID controller:

$$G_{PID}(s) = K_p\left(1 + T_d s + \frac{1}{T_i s}\right), \tag{3.2b}$$

3.1. Slow-rate PI controller

The following two cases must be considered:

a) No dropout ($d_k^{lr} = 1$): The PI controller working at period *NT* is enunciated as

$$G_{PI}^{NT}(z_N) = K_{PI}\,\frac{z_N - \left(1 - \dfrac{NT}{T_i}\right)}{z_N - 1} = \frac{U_{PI}^{NT}(z_N)}{E_{PI}^{NT}(z_N)}, \tag{3.3}$$

being $U_{PI}^{NT}(z_N)$ the PI control signal, $E_{PI}^{NT}(z_N)$ the error signal, and K_{PI}, T_i the gains of the PI controller (usually, $K_{PI} = 1$) . The PI control signal is obtained as

$$U_{PI}^{NT}(z_N) = G_{PI}^{NT}(z_N)E_{PI}^{NT}(z_N) = G_{PI}^{NT}(z_N)\left(R^{NT}(z_N) - Y^{NT}(z_N)\right), \tag{3.4}$$

and, from (3.3), the difference equation for the PI controller (with $K_{PI} = 1$) will be

$$u_{PI,k}^{NT} = u_{PI,k-1}^{NT} + e_{PI,k}^{NT} - \left(1 - \frac{NT}{T_i}\right)e_{PI,k-1}^{NT} = u_{PI,k-1}^{NT} + \left(r_k^{NT} - y_k^{NT}\right) - \left(1 - \frac{NT}{T_i}\right)\left(r_{k-1}^{NT} - y_{k-1}^{NT}\right), \tag{3.5}$$

b) Dropout ($d_k^{lr} = 0$): In this case, instead of using the actual PI control signal in (3.4), the estimated one $\hat{U}_{PI}^{NT}(z_N)$ must be used. This signal is previously generated at the prediction stage according to subsection 3.4.

3.2. Rate converter

As it is well-known [17], a rate converter $\left[H^{NT}\right]^T(\bar{z})$ between slow (remote) and fast (local) controllers is required. Its goal is to convert the slow rate PI control signal to a fast rate one in order to be used as an input to the fast rate PD controller. This operation can be carried out either at the remote side (sending the converted signal to the local side) or directly at the local side (this is the option used in this work). Two cases are considered:

a) No dropout ($d_k^{rl}=1$): The rate converter considers the actual slow-rate PI control signal $U_{PI}^{NT}(z_N)$ to obtain the held fast-rate one $\bar{U}_{PI}^{T}(\bar{z})$. As, in this work, step references are considered, the rate converter becomes a digital zero order hold:

$$\left[H^{NT}\right]^{T}(\bar{z})=\frac{\bar{U}_{PI}^{T}(\bar{z})}{\left[U_{PI}^{NT}(z_N)\right]^{T}}=\frac{1-\bar{z}^{-N}}{1-\bar{z}^{-1}}\rightarrow\bar{U}_{PI}(\bar{z})=\left[H^{NT}\right]^{T}\left[U_{PI}^{NT}(z_N)\right]^{T}, \tag{3.6a}$$

Note that $U_{PI}^{NT}(z_N)$ is required to be used in an expanded way $\left[U_{PI}^{NT}(z_N)\right]^{T}$, that is,

$$\left[U_{PI}^{NT}(z_N)\right]^{T}\triangleq\tilde{U}_{PI}^{T}(\bar{z})\triangleq\sum_{k=0}^{\infty}\tilde{u}_{PI,k}^{T}\bar{z}^{-k}:\begin{cases}\tilde{u}_{PI,k}^{T}=u_{PI,k}^{T},\forall k=\lambda N\\ \tilde{u}_{PI,k}^{T}=0, \quad \forall k\neq\lambda N\end{cases},\lambda\in Z^{+}, \tag{3.6b}$$

More information can be found in [17].

b) Dropout($d_k^{rl}=0$): Now, the rate converter considers the estimated PI control signal $\hat{U}_{PI}^{NT}(z_N)$, that is

$$\hat{\bar{U}}_{PI}(\bar{z})=\left[H^{NT}\right]^{T}\left[\hat{U}_{PI}^{NT}(z_N)\right]^{T}, \tag{3.7}$$

As it was said in section 2, for the sake of simplicity and brevity, both cases ((3.6a) and (3.7)) will be contained under the notation $\dot{\bar{U}}_{PI}^{T}(\bar{z})$ from now on.

3.3. Fast-rate PD controller

a) No dropout ($d_k^{rl}=1$)

After (2.1), the round-trip time delay was defined to fulfill $\tau_k\in\left[0,\tau_{\max}\right]$. Let us suppose $\tau_{\max}<T$ (for example, $\tau_{\max}=T-t$) in order to inject *N* control actions for each sensor period *NT* (remember Figure 3). The PD controller, working at period *T*, is enunciated as

$$G_{PD}^{T,\tau_k}(\bar{z})=K_{PD}^{\tau_k}\frac{\bar{z}\left(1+\frac{T_d^{\tau_k}}{T}\right)-\frac{T_d^{\tau_k}}{T}}{\bar{z}}=\frac{\dot{U}_{PD}^{T,\tau_k}(\bar{z})}{\dot{\bar{U}}_{PI}^{T}(\bar{z})}\rightarrow\dot{U}_{PD}^{T,\tau_k}(\bar{z})=G_{PD}^{T,\tau_k}(\bar{z})\dot{\bar{U}}_{PI}^{T}(\bar{z}), \tag{3.8}$$

where $K_{PD}^{\tau_k},T_d^{\tau_k}$ are the gains of the PD controller, which are retuned according to τ_k via the gain-scheduling algorithm presented in [20]. Note that the notation $\dot{U}_{PD}^{T,\tau_k}(\bar{z})$ represents the PD control signal obtained either from the actual PI control signal $\bar{U}_{PI}^{T}(\bar{z})$ (in (3.6a)) or from its estimation $\hat{\bar{U}}_{PI}^{T}(\bar{z})$ (in (3.7)). At the current sensor period, in addition to the last PD control action of the previous period (which remains held), new *N* PD control actions are applied to the plant after τ_k (remember Figure 3). These *N* actions are obtained after iterating the difference equation deduced from (3.8) *N* times, that is

$$\dot{u}_{PD,k}^{T,\tau_k} = K_{PD}^{\tau_k}\left(1+\frac{T_d^{\tau_k}}{T}\right)\dot{\bar{u}}_{PI,k}^{T} - K_{PD}^{\tau_k}\left(\frac{T_d^{\tau_k}}{T}\right)\dot{\bar{u}}_{PI,k-1}^{T}, \tag{3.9}$$

As commented in section 2, due to the delay, these actions will be applied following a non-uniform pattern. Then, a basic period *t* is required to adapt the non-uniformity to the delay in such a way that the actuation pattern inside the sensor period *NT* will take this form (where *l*=0..*LN*-1):

$$\begin{cases} \dot{u}_{PD,k-1}^{T,\tau_k}, & lt = 0..\tau_k \\ \dot{u}_{PD,k}^{T,\tau_k}, & lt = \tau_k..T \\ \dot{u}_{PD,k+1}^{T,\tau_k}, & lt = T..2T \\ \vdots & \\ \dot{u}_{PD,k+N-1}^{T,\tau_k}, & lt = (N-1)T..NT \end{cases}, \tag{3.10}$$

b) Dropout ($d_k^{rl} = 0$): The estimated PI control signal $\hat{\bar{U}}_{PI}^{T}(\bar{z})$ (to be defined in the last step in subsection 3.4) is now required. This control signal is available at the local side, since it was calculated at the remote side in a previous iteration and sent to the local side in a previous successful communication.

The time $\tau_{\max} = T - t$ is considered as the maximum waiting time established to detect a packet dropout (remember Figure 4). In this way, *N* PD control actions (in addition to the held action from the previous sensor period) are guaranteed to be applied in the current sensor period (as in (3.10)). In this case, with certainty, the PD control signal is an estimated one, since it depends on the estimated PI control signal. Then, with $\tau_k = \tau_{\max}$

$$\hat{U}_{PD}^{T,\tau_k}(\overline{z}) = G_{PD}^{T,\tau_k}(\overline{z})\hat{\overline{U}}_{PI}^{T}(\overline{z}), \tag{3.11}$$

From (3.9), but considering estimated signals, the set of *N* control actions can be computed and applied according to the next non-uniform actuation pattern inside the sensor period *NT* (where *t* is the basic period, and *l*=0..*LN*-1):

$$\begin{cases} \dot{u}_{PD,k-1}^{T,\tau_k}, & lt = 0..\tau_k \\ \hat{u}_{PD,k}^{T,\tau_k}, & lt = \tau_k..T \\ \hat{u}_{PD,k+1}^{T,\tau_k}, & lt = T..2T \\ \vdots & \\ \hat{u}_{PD,k+N-1}^{T,\tau_k}, & lt = (N-1)T..NT \end{cases}, \tag{3.12}$$

3.4. Prediction stage

The prediction algorithm is executed *M* times (*M* is the upper bound of consecutive packet dropouts) following a cascade structure (remember Figure 2) in order to generate the packet which includes the future, estimated *M* PI control actions $\left[\hat{u}_{PI,k+1}^{NT}, \hat{u}_{PI,k+2}^{NT}, \ldots \hat{u}_{PI,k+M}^{NT}\right]$. This packet is computed for every sensor period at the remote side, and it is sent to the local side in order to be stored, and used if subsequent dropouts occur through the remote-to-local communication. Considering a for-loop where *i*=1..*M*, the statements of the prediction algorithm included in the loop are based on the next steps:

1. Resetting of the initial state: If the current state sensed at period *NT*, x_k^{NT}, is available at the remote side (that is, no dropout occurs when being sent via the local-to-remote network link), a resetting of the initial condition for the state at period *t* and *T* can be carried out. This operation can be executed when *i*=1, and it is required in order to deal with unstable plants or marginally stable plants [23, 24] such as the one used in this work. For the rest of iterations of the algorithm (*i*=2..*M*), or if the current state was dropped (for *i*=1), the updating is computed from the estimated state $\hat{x}_{k+i-1}^{NT}$ (to be defined in step 3). As in section 2, to contemplate every situation, let us define a generic (actual or estimated) state $\dot{x}_k^{NT}$. Therefore, the resetting carried out in each iteration is

$$\begin{cases} i=1: & \hat{x}_k^T \leftarrow \dot{x}_k^{NT}; \quad \hat{x}_k^t \leftarrow \dot{x}_k^{NT} \\ i>1: & \hat{x}_{k+(i-1)N}^T \leftarrow \hat{x}_{k+i-1}^{NT}; \quad \hat{x}_{k+(i-1)LN}^t \leftarrow \hat{x}_{k+i-1}^{NT} \end{cases}, \tag{3.13}$$

2. Estimation of the N PD control actions either from the estimated PI control signal $\hat{\bar{U}}_{PI}^T(\bar{z})$ (it can occur for $i \geq 1$) or from the actual one $\bar{U}_{PI}^T(\bar{z})$ (it can only occur for i=1). Both cases assume the previous rate conversion ((3.7) or (3.6a), respectively).

   Similarly to (3.9), the estimated control signal is computed by iterating the next difference equation for j=0..N-1. Each iteration i for the prediction algorithm is calculated as follows

$$\begin{cases} i=1: & \hat{u}_{PD,k+j}^{T,\tau_k} = K_{PD}^{\tau_k}\left(1+\frac{T_d^{\tau_k}}{T}\right)\dot{\bar{u}}_{PI,k+j}^T - K_{PD}^{\tau_k}\left(\frac{T_d^{\tau_k}}{T}\right)\dot{\bar{u}}_{PI,k-1+j}^T, \quad \tau_k=\tau_m \\ i>1: & \hat{u}_{PD,k+j+(i-1)N}^{T,\tau_k} = K_{PD}^{\tau_k}\left(1+\frac{T_d^{\tau_k}}{T}\right)\hat{\bar{u}}_{PI,k+j+(i-1)N}^T - K_{PD}^{\tau_k}\left(\frac{T_d^{\tau_k}}{T}\right)\hat{\bar{u}}_{PI,k-1+j+(i-1)N}^T, \quad \tau_k=\tau_{\max} \end{cases}, \tag{3.14}$$

   where now, the equation for the first iteration of the prediction algorithm (i=1) is calculated supposing a successful remote-to-local communication (the packet which includes the estimated PI control actions will arrive to the local side) but unknowing the consequent remote-to-local delay (this information is not available at this moment) and hence, the round-trip time delay. This is the reason of adopting the statistical mode of the delay distribution τ_m as the delay to be considered at the first iteration. The mode can be obtained via a previous statistical analysis about the delay nature. For the rest of iterations (when i>1), the considered delay to generate the PD control actions is the maximum waiting time established to detect a dropout, that is, $\tau_k = \tau_{\max}$, since the next M-1 packets are assumed to be dropped.

3. Estimation of the next state and output at period NT from the estimated PD control actions. As in (3.10) and (3.11), the basic period t is used. In this case, for each iteration of the prediction algorithm i=1..M, the next state-space representation is calculated by iterating for l=0..LN-1

$$\begin{cases} \hat{x}^t_{k+1+l+(i-1)LN} = A^t \hat{x}^t_{k+l+(i-1)LN} + B^t \left\langle \hat{u}^{T,\tau_k}_{\cdot\cdot PD,k+l+(i-1)LN} \right\rangle^t \\ \hat{y}^t_{k+1+l+(i-1)LN} = C^t \hat{x}^t_{k+1+l+(i-1)LN} \end{cases}, \tag{3.15}$$

where, for simplicity and brevity, let us represent by means of $\left\langle \hat{u}^{T,\tau_k}_{\cdot\cdot PD,k+l+(i-1)LN} \right\rangle^t$ the sequence of the N+1 PD control actions included in each sensor period by holding them at period t, that is, applying them via a non-uniform actuation pattern such as the one used in (3.12). As a result of iterating (3.15) for all i, the M estimated states and outputs at period NT, $\hat{x}^{NT}_{k+i}, \hat{y}^{NT}_{k+i}$, are obtained.

4. Estimation of the PI control signal $\hat{U}^{NT}_{PI}(z_N)$ from the estimated output signal $\hat{Y}^{NT}(z_N)$. Note that, particularly for the first iteration of the prediction algorithm (i=1), the actual output y^{NT}_k can be used if it is available at the remote side, that is, if no dropout occurs when being sent through the local-to-remote network link ($d^{lr}_k = 1$). Then, the actual PI control action $u^{NT}_{PI,k}$, which is generated by the output y^{NT}_k (remember (3.5)), can also be used. In this way, similarly to step 1, a resetting of the initial condition for the PI controller ($u^{NT}_{PI,k}$) is carried out in order to compute the next estimated PI control action $\hat{u}^{NT}_{PI,k+1}$. This operation is useful due to the marginally stable open-loop nature of the PI controller [23, 24]. As usual, in order to contemplate every situation in the prediction algorithm, let us define a generic (actual or estimated) output $\dot{y}^{NT}_k$, and a generic (actual or estimated) control action $\dot{u}^{NT}_{PI,k}$. Therefore, similarly to (3.5), the iterations i of the prediction algorithm take the form

$$\begin{cases} i=1: \quad \hat{u}^{NT}_{PI,k+1} = \dot{u}^{NT}_{PI,k} + \left(r^{NT}_{k+1} - \hat{y}^{NT}_{k+1}\right) - \left(1 - \dfrac{NT}{T_i}\right)\left(r^{NT}_k - \dot{y}^{NT}_k\right) \\ i>1: \quad \hat{u}^{NT}_{PI,k+i} = \hat{u}^{NT}_{PI,k+i-1} + \left(r^{NT}_{k+i} - \hat{y}^{NT}_{k+i}\right) - \left(1 - \dfrac{NT}{T_i}\right)\left(r^{NT}_{k+i-1} - \hat{y}^{NT}_{k+i-1}\right) \end{cases}, \tag{3.16}$$

4. Stability analysis

4.1 Closed-loop model via lifting

Let us consider a continuous linear time-invariant plant, which admits a state-space realization $\Sigma = (\bar{A}, \bar{B}, C, D)$, with suitable dimensions. Being ξ an arbitrary real number, one can denote $B(\xi) = \int_0^{\xi} e^{\bar{A}\gamma} \bar{B} d\gamma$ if $\xi > 0$, or $B(\xi) = 0$ if $\xi \leq 0$. The discrete time sampled-data model at period *T* of the previous plant was presented in (3.2a), being $A^T = e^{\bar{A}T}; \quad B^T = B(T); \quad C^T = C$.

In order to reflect the dual-rate sampling with non-uniform actuation (inputs are applied at instants $\lambda_i \left(i = 0,1,\ldots,N \right)$; concretely, as detailed in Section 2, the actuation pattern inside a sensor period is $\{0, \tau_k, T, 2T, \ldots, (N-1)T\}$, and hence $\lambda_0 = 0; \lambda_1 = \tau_k; \lambda_2 = T; \ldots; \lambda_N = (N-1)T$), the control system can be modeled via state-space representation adopting the so-called lifting methodology [26]. Then, let us represent the process as:

$$\begin{aligned} x_{k+1}^{NT} &= A_p x_k^{NT} + B_p U_{PD}^{T,\tau_k} \\ y_k^{NT} &= C_p x_k^{NT} \end{aligned} , \tag{4.1}$$

where

- $A_p = \begin{pmatrix} e^{\bar{A}NT} & B_0^* \\ 0 & 0 \end{pmatrix}; \quad B_p = \begin{pmatrix} B_1^* & \cdots & B_N^* \\ 0 & \cdots & 1 \end{pmatrix}; \quad C_p = \begin{pmatrix} C & 0 \end{pmatrix}; \quad$ being $B_i^* = B(\lambda_{i+1} - \lambda_i) e^{\bar{A}(NT - \lambda_{i+1})}$.
- $U_{PD}^{T,\tau_k} = \left[u_{PD,k}^{T,\tau_k} \quad u_{PD,k+1}^{T,\tau_k} \quad \ldots \quad u_{PD,k+N-1}^{T,\tau_k} \right]^{\text{tr}}$ is a generic array of *N* control signals ($\left(\bullet \right)^{\text{tr}}$ means transpose function) to indistinctly represent the *N* computed actions that appear both in (3.10) and in (3.12), that is, all the actions included in these equations except the last action from the previous sensing period, $u_{PD,k-1}^{T,\tau_k}$. This action $u_{PD,k-1}^{T,\tau_k}$ is included in the state x_k^{NT} as a state variable which is memorised from one sensing period to the next one, such as in [18], equation (13).

The predictor stage can be modelled as follows:

$$\hat{x}_{k+1}^{NT} = \begin{cases} A_p \hat{x}_k^{NT} + B_p U_{PD}^{T,\tau_k}, & if\ d_k^{lr} = 0 \\ A_p x_k^{NT} + B_p U_{PD}^{T,\tau_k}, & if\ d_k^{lr} = 1 \end{cases}, \tag{4.2}$$

For the sake of simplicity and brevity, let us consider the following assumptions to define the state-space representation of the delay-dependent controller:

1) if setpoints are constant, we can assume them to be zero without loss of generality, and hence $e_k^{NT} = -y_k^{NT}$

2) the control system behaviour when $d_k^{lr} = 0$ is similar to the one when $d_k^{rl} = 0$ from the point of view of computing the control signal U_{PD}^{T,τ_k} from the estimated output $\hat{y}_k^{NT}$. That is, when $d_k^{rl} = 0$, the control signal U_{PD}^{T,τ_k} to be applied at iteration *k* is generated from a PI control action which was previously computed at the predictor stage taking into account an output estimation for the time instant *k*, $\hat{y}_k^{NT}$. When $d_k^{lr} = 0$, obviously the controller needs $\hat{y}_k^{NT}$ to calculate U_{PD}^{T,τ_k}.

3) the dual-rate controller can be defined as a cascade-connected system.

After manipulating the difference equations (3.5) and (3.9), and assuming such as in [18] (Section 5) that the proportional and integral actions are generated at the slow period, and the derivative one (which has relationship with anticipation and high-frequency behavior) is concentrated in the first sample, the dual-rate controller can take this lifted expression

$$\begin{matrix} \begin{cases} \varphi_{k+1}^{NT} = A_c \varphi_k^{NT} - B_c \hat{y}_{k+1}^{NT} + \overline{B}_c \hat{y}_k^{NT} \\ U_{PD}^{T,\tau_k} = C_c(\tau_k)\varphi_k^{NT} \end{cases}, & if\ d_k^{lr} = 0\ or\ d_k^{rl} = 0 \\ \begin{cases} \varphi_{k+1}^{NT} = A_c \varphi_k^{NT} - B_c y_{k+1}^{NT} + \overline{B}_c y_k^{NT} \\ U_{PD}^{T,\tau_k} = C_c(\tau_k)\varphi_k^{NT} \end{cases}, & if\ d_k^{lr} = d_k^{rl} = 1 \end{matrix}, \tag{4.3}$$

where $\varphi_k^{NT} = \begin{pmatrix} u_{PI,k}^{NT} & \mu_k^{NT} \end{pmatrix}^{\mathrm{tr}}$, being $\mu_k^{NT} = u_{PI,k-1}^{NT}$, and

$$
\begin{aligned}
&A_c = \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix}; \quad B_c = \begin{pmatrix} 1 \\ 0 \end{pmatrix}; \quad \bar{B}_c = \begin{pmatrix} 1 - \dfrac{NT}{T_i} \\ 0 \end{pmatrix}; \\
&C_c(\tau_k) = \begin{pmatrix} K_{PD}^{\tau_k}\left(1 + \dfrac{T_d^{\tau_k}}{T}\right) & -K_{PD}^{\tau_k}\dfrac{T_d^{\tau_k}}{T} \\ K_{PD}^{\tau_k} & 0 \\ \vdots & \vdots \\ K_{PD}^{\tau_k} & 0 \end{pmatrix} \qquad ,
\end{aligned}
\tag{4.4}
$$

To obtain the closed-loop model, the following definition is used

$$
v_k^{NT} = y_k^{NT} - \hat{y}_k^{NT} = C_p(x_k^{NT} - \hat{x}_k^{NT}), \tag{4.5}
$$

and this state variable is adopted $\bar{x}_k^{NT} = \begin{pmatrix} x_k^{NT} & \varphi_k^{NT} & v_k^{NT} \end{pmatrix}^{\mathrm{tr}}$. Some manipulations lead to the closed-loop model:

$$
\bar{x}_{k+1}^{NT} = \begin{cases} A_{cl,0}(\tau_k)\bar{x}_k^{NT}, & if\ d_k^{lr} = 0\ or\ d_k^{rl} = 0 \\ A_{cl,1}(\tau_k)\bar{x}_k^{NT}, & if\ d_k^{lr} = d_k^{rl} = 1 \end{cases} \qquad , \tag{4.6}
$$

where

$$
\begin{aligned}
&A_{cl,0}(\tau_k) = \begin{bmatrix} A_p & B_p C_c(\tau_k) & 0 \\ \bar{B}_c C_p - B_c C_p A_p & A_c - B_c C_p B_p C_c(\tau_k) & -\bar{B}_c - A_p B_c \\ 0 & 0 & A_p \end{bmatrix}, \\
&A_{cl,1}(\tau_k) = \begin{bmatrix} A_p & B_p C_c(\tau_k) & 0 \\ \bar{B}_c C_p - B_c C_p A_p & A_c - B_c C_p B_p C_c(\tau_k) & 0 \\ 0 & 0 & 0 \end{bmatrix}
\end{aligned}
\tag{4.7}
$$

4.2 Closed-loop stability

Similarly to [9,19], the next Theorem can be enunciated to assess the closed-loop system stability.

Theorem: Given $P[\tau_k]$ in (2.2), the closed-loop system in (4.6) is stable if there exists a positive definitive solution $Q = Q^{\mathrm{tr}} > 0$ for the following probabilistic LMIs

$$\begin{array}{l}\sum_{j=1}^{l} P\left[\vartheta_j\right] A_{cl,0}^{\mathrm{tr}}(\vartheta_j)\cdot Q\cdot A_{cl,0}(\vartheta_j)-Q<0 \\ \sum_{j=1}^{l} P\left[\vartheta_j\right] A_{cl,1}^{\mathrm{tr}}(\vartheta_j)\cdot Q\cdot A_{cl,1}(\vartheta_j)-Q<0\end{array}, \qquad (4.8)$$

where ϑ is a dummy parameter ranging in a set Θ where τ_k is assumed to take values in, being Θ an interval $\left[0,\tau_{\max}\right]$, and l the number of equally spaced values to get a dense enough gridding in ϑ. (4.8) can be solved by means of widely known methods [27].

Proof: Let $V_k^{NT}=\left(\overline{x}_k^{NT}\right)^{\mathrm{tr}}\cdot Q\cdot\overline{x}_k^{NT}$ be a Lyapunov candidate. Taking $E\left[V_k^{NT}\right]$ as the statistical expectation for the Lyapunov function, and assuming a probabilistic LMI gridding procedure, then the expectation of the increment $E\left[\Delta V_k^{NT}\right]$ along subsystem $A_{cl,0}(\tau_k)$ in (4.6) can be obtained as follows

$$\begin{aligned} & E\left[\Delta V_k^{NT}\right]=E\left[V_{k+1}^{NT}-V_k^{NT}\right]=E\left[\left(\overline{x}_{k+1}^{NT}\right)^{\mathrm{tr}}\cdot Q\cdot\overline{x}_{k+1}^{NT}-\left(\overline{x}_k^{NT}\right)^{\mathrm{tr}}\cdot Q\cdot\overline{x}_k^{NT}\right]= \\ & =E\left[\left(A_{cl,0}(\tau_k)\cdot\overline{x}_k^{NT}\right)^{\mathrm{tr}}\cdot Q\cdot A_{cl,0}(\tau_k)\cdot\overline{x}_k^{NT}-\left(\overline{x}_k^{NT}\right)^{\mathrm{tr}}\cdot Q\cdot\overline{x}_k^{NT}\right]= \\ & =E\left[\left(\overline{x}_k^{NT}\right)^{\mathrm{tr}} A_{cl,0}^{\mathrm{tr}}(\tau_k)\cdot Q\cdot A_{cl,0}(\tau_k)\cdot\overline{x}_k^{NT}-\left(\overline{x}_k^{NT}\right)^{\mathrm{tr}}\cdot Q\cdot\overline{x}_k^{NT}\right]= \\ & =E\left[\left(\overline{x}_k^{NT}\right)^{\mathrm{tr}}\left(A_{cl,0}^{\mathrm{tr}}(\tau_k)\cdot Q\cdot A_{cl,0}(\tau_k)-Q\right)\cdot\overline{x}_k^{NT}\right]= \\ & =\left(\overline{x}_k^{NT}\right)^{\mathrm{tr}}\left(\sum_{j=1}^{l} P\left[\vartheta_j\right] A_{cl,0}^{\mathrm{tr}}(\vartheta_j)\cdot Q\cdot A_{cl,0}(\vartheta_j)-Q\right)\overline{x}_k^{NT}<0 \end{aligned}, \qquad (4.9)$$

A similar development for subsystem $A_{cl,1}(\tau_k)$ leads to (4.8).

5. Simulation results

5.1. Simulation data

In this section, a particular case for the proposed NCS is presented. The control solution is shown by simulation, which is based on a model of a plant available in the laboratory in order to then validate the results experimentally (in section 6).

The process selected to be controlled is a Cartesian robot manufactured by Inteco, specifically, the 3D CRANE module (see in Figure 5). The rail measures of this plant for each axis are: X=0.050m, Y=0.040m, Z=0.050m.

Focusing on the X axis, it is identified, obtaining the next model

$$G_p(s) = \frac{6.3}{s(s+17.7)} \text{ m/c.a.u.}, \tag{5.1}$$

where c.a.u. means control action units, which are generated by a PWM signal normalized in the range [0,1].

The system also presents two non-linear behaviors to be taken into account in real-time implementation: saturation limits of control actions in ±1, and dead zone values of ±0.06. Both of them are identified experimentally and measured in normalized c.a.u.

In [20], a conventional PID controller such as in (3.2b) is used with $K_p = 12$, $T_d = 0.01$ and $T_i = 3.5$, which reaches certain specifications (null steady-state error, settling time around 4s, and overshot around 5%). Digitally implementing this controller at a period higher than 0.1s, the aforementioned specifications cannot be assured. It is assumed that the sensor's nature or the network load do not allow a sampling period below 0.2s. Therefore, a sample time of *NT*=0.2s is used, and a dual-rate controller with *N*=2 is implemented using (3.3)-(3.12) (assuming $K_{PD} = K_p$ and $K_{PI} = 1$) in order to reach the specifications. The following gain-scheduling law for the dual-rate delay-dependent controller is used to deal with the round-trip time delay τ_k:

$$\begin{aligned} K_{PD}^{\tau_k} &= -50\tau_k + K_{PD} \\ T_d^{\tau_k} &= 0.5\tau_k + T_d \end{aligned}, \tag{5.2}$$

For the present study, as enunciated in (2.2), a generalized exponential distribution is considered. The histogram used in this case is shown in Figure 6, where τ_k takes values in the range $\Theta = [0.04, 0.08)$. In addition, as presented in (2.3), packet dropouts are modeled as a Bernoulli distribution, being in this case $p = p^{lr} = p^{rl} = 0.3$ and *M*=3.

In order to assess the stability of the setup in probabilistic time-varying delays, the LMI gridding in (4.8) has been carried out taking l=20 grid points so as to compute the closed-loop realizations $A_{cl,0}(\vartheta_j)$ and $A_{cl,1}(\vartheta_j)$ for the parameter space Θ. From Figure 6, $P\left[\vartheta_j\right]$ is normalized in order to take values in $\left[0,1\right)$. As a result, it can be checked that a feasible solution Q exists, which guarantees stability for the control system design:

$$Q = 10^6 \begin{bmatrix} 1.20050 & 0.03926 & -0.11557 & -0.13543 & -0.01666 & 0.27511 & 0 & 0.00041 \\ 0.03926 & 2.30974 & 0.00751 & 0.80299 & -0.00905 & -0.26529 & 0 & -0.00046 \\ -0.11557 & 0.00751 & 0.04032 & -0.06609 & -0.11587 & 0.02679 & 0 & 0.00003 \\ -0.13543 & 0.80299 & -0.06609 & 2.53137 & -0.01202 & -0.68629 & 0 & -0.00036 \\ -0.01666 & -0.00905 & -0.11587 & -0.01202 & 1.13465 & -0.02103 & 0 & -0.00004 \\ 0.27511 & -0.26529 & 0.02679 & -0.68629 & -0.02103 & 3.08272 & 0.06604 & 0.00297 \\ 0 & 0 & 0 & 0 & 0 & 0.06604 & 1.16892 & 0.00639 \\ 0.00041 & -0.00046 & 0.00003 & -0.00036 & -0.00004 & 0.00297 & 0.00639 & 1.15074 \end{bmatrix}, \quad (5.3)$$

Figure 7 shows the results obtained for the control solution, where filtered step references are used in order to avoid the saturation of the control signal. Note that the sequence of packet dropouts is represented in the time axis in such a way that a point indicates a packet dropout in the time instant where it is plotted. If the point increases its value in the vertical axis, then consecutive dropouts are occurring in this instant. Figure 7 also shows the desired, nominal (no-delay, no-dropout) output. When packet dropouts appear in the NCS, if the delay-dependent control solution does not include the prediction stage, behavior deterioration is observed. Then, including the prediction stage, the control performance can be restored at about the same level as the nominal one.

5.2. Data anylisis via cost indexes

In this subsection four different cost indexes will be used. Firstly, in order to better quantify the benefits of the packet-based control solution, the cost indexes J_1 and J_2 are utilized. The first one, J_1, is based on the Integral of Absolute Error (IAE), and the second one, J_2, on the overshoot value. J_1 and J_2 take the worst behavior, that is, the delay-dependent controller with no prediction stage as the reference value to be compared to the rest of behaviors in order to compute the consequent improvements.

In order to define J_1 let us consider the array Y, which includes the different sequence of outputs to be analyzed for the dual-rate control system, that is, $Y=\left[Y_{Nom},Y_{DD-NP},Y_{DD-P}\right]$, being Y_{Nom} the output for the nominal (no-delay, no-dropout) case, Y_{DD-NP} the output for the delay-dependent controller with no prediction stage, and Y_{DD-P} the output for the delay-dependent controller with prediction stage.

From Y the following accumulated (integrated) error E_Y in a range of time instants Γ can be computed

$$E_Y(i)=\sum_{\Gamma}\left|Y(i)-Y_{Nom}\right|, \quad i=1..3, \tag{5.4}$$

Then, the J_1 cost index takes this form

$$J_1(i)=100-\frac{E_Y(i)}{E_Y(2)}100\,(\%), \quad i=1..3, \tag{5.5}$$

being $E_Y(2)$ the worst expected accumulated error, that is, the error reached by the delay-dependent controller with no prediction stage. Therefore, the rest of the errors are measured by J_1 as an improvement (in %) with regard to $E_Y(2)$.

In order to define J_2, from Y the following overshoot O_Y in a range of time instants Γ can be calculated (considering positive -max- or negative -min- filtered step references)

$$O_Y(i)=\max\left(\left|\max_{\Gamma}Y(i)-\max_{\Gamma}Y_{Nom}\right|,\left|\min_{\Gamma}Y(i)-\min_{\Gamma}Y_{Nom}\right|\right), \quad i=1..3, \tag{5.6}$$

Then, the J_2 index is

$$J_2(i)=100-\frac{O_Y(i)}{O_Y(2)}100\,(\%), \quad i=1..3, \tag{5.7}$$

being $O_Y(2)$ the worst expected overshoot, that is, the overshoot reached by the delay-dependent controller with no prediction stage. Similarly to J_1, the rest of the overshoots are measured by J_2 as an improvement (in %) with regard to $O_Y(2)$.

Table 1 summarizes the cost indexes obtained for each output. As expected, considering packet dropouts, the packet-based control strategy which includes prediction stage clearly improves the control performance obtained by the no-prediction control strategy.

6. - Experimental results

To validate the simulation results obtained in section 5, a laboratory test-bed platform is set up, which includes the CRANE module previously presented, two computers and an Ethernet cable. One computer is directly connected to the plant and composes the local part of the control system. The aims of this computer are: firstly, to be in charge of the measures on the plant at *NT*=0.2s and their communication; secondly, to be responsible for the reception of the slow-rate control signal from the PI controller, the calculation of the fast-rate PD control actions (at *T*=0.1s) and their injection over the plant. When convenient, it is also in charge of the reception and selection of signals predicted to be applied to the plant when loss of communication occurs.

The second computer performs the remote part of the controller, receiving the outputs of the plant, calculating the slow-rate PI controller, and sending back these actions to the local system. When required, this part is also in charge of the calculation of future, slow, predicted control actions, which will be sent together with the slow-rate control signal.

These computers are connected by a UDP network through an Ethernet cable that performs the local-to-remote and remote-to-local links. In order to obtain similar conditions to those considered in simulation, network delays and packet dropouts are modified by software.

Figure 8 shows the outputs obtained in the experiment, which clearly reveals the same trend observed in Figure 7. To better validate the results, Table 2 details the cost indexes J_1 and J_2

computed for the experiment, where lower improvements than in simulation are achieved due to practical issues (possible divergences between model and plant, dead zone, and so on).

7. - Conclusions

In this work, a packet-based dual-rate control solution is presented for an NCS where time-varying delays, packet dropouts and packet disorder can occur. Selecting the sensing period greater than the longest round-trip time delay, the packet disorder is avoided. In addition, energy saving can be achieved by reducing network load, which is a crucial issue, especially in battery-powered wireless sensors. However, in order to reach certain specifications, an N times faster actuation period must be used, leading to the dual-rate control structure. The time-varying delays are dealt with the gain scheduling approach in [20]. By means of the packet-based control strategy, and including a prediction stage which is able to treat unstable or marginally stable plants, the packet dropout problem is solved. Control system stability is ensured via probabilistic LMIs.

Simulation results reveal the improvements reached by the control proposal, which are validated by means of a laboratory test-based platform, where the process to be controlled is a Cartesian robot.

References

1. Zhang, X-M.; Han, Q-L.; Yu, X. Survey on recent advances in networked control systems. *IEEE Transactions on Industrial Informatics* **2016**, vol.12, n.5, pp.1740-1752.
2. Zhang, D.; Shi, P.; Wang, Q.G.; Yu, L. Analysis and synthesis of networked control systems: a survey of recent advances and challenges. *ISA transactions* **2017**, vol.66, pp.376-392.
3. Qiu, J.; Gao, H.; Chow, M-Y. Networked control and industrial applications [Special section introduction]. *IEEE Transactions on Industrial Electronics* **2016**, vol.63, n.2, p.1203-1206.
4. Rahmani, B.; Markazi, A.; Behzad. S. A new method for control of networked systems with an experimental verification. *ISA Transactions* **2015**, vol.56, n.1, pp.299-307.

5. Liang, G. Control and communication co-design: Analysis and practice on performance improvement in distributed measurement and control system based on fieldbus and Ethernet. *ISA Transactions* **2015**, vol.54, pp.169-192. doi: 10.1016/j.isatra.2014.08.012

6. Santos, C.; Martínez-Rey, M.; Espinosa, F.; Gardel, A.; Santiso, E. Event-Based Sensing and Control for Remote Robot Guidance: An Experimental Case. *Sensors*, **2017**, vol.17, n.9, p.2034.

7. Pizá, R.; Salt, J.; Sala, A.; Cuenca, A. Hierarchical Triple-Maglev Dual-Rate Control Over a Profibus-DP Network, *IEEE Trans. on Control Systems Technology* **2014**, vol.22, n.1, pp.1-12.

8. Georgiev, D.; Tilbury, D.M. Packet-based control: The H2-optimal solution. *Automatica* **2006**, vol.42, n.1, pp.137-144.

9. Zhao, Y.; Liu, G.; Rees, D. Design of a packed-based control framework for networked control systems, *IEEE Trans. Control Syst. Technol.* **2009**, vol.17, n.5, pp.859-865.

10. Zhao, Y.; Liu, G.; Rees, D. Modelling and stabilization of continuous-time packed-based networked control systems, *IEEE Trans. on Systems, Man, and Cybernetics – Part B: Cybernetics* **2009**, vol.39, n.6, pp.1646-1652.

11. Zhao, Y.; Liu, G.; Kang, Y.; Yu, L. Packet-Based Control Design for Networked Control Systems. In *Packet-Based Control for Networked Control Systems* **2017**, pp.15-32. Springer.

12. Heemels, W. P. M. H.; Donkers, M. C. F.; Teel, A. Periodic event-triggered control for linear systems, *IEEE Trans. on Automatic Control* **2013**, vol.58, n.4, pp. 698-711.

13. Villarreal-Cervantes, M. G.; Guerrero-Castellanos, J.F.; Ramírez-Martínez, S.; Sánchez-Santana, J. P. Stabilization of a (3, 0) mobile robot by means of an event-triggered control. *ISA Trans.* **2015**. vol.58 pp.605-613.

14. Casanova, V.; Alcaína, J.; Salt, J.; Pizá, R.; Cuenca, A. Control of the rotary inverted pendulum through threshold-based communication. *ISA Transactions* **2016**. vol.62 pp.357-366.

15. Lee, J-H.; Jung, I-B. Adaptive-Compression Based Congestion Control Technique for Wireless Sensor Networks. *Sensors* **2010**, vol.10, n.4, pp.2919-2945, doi:10.3390/s100402919.

16. Chen, W.; Qiu, L. Stabilization of networked control systems with multirate sampling. *Automatica* **2013**, vol.49, n.6, pp.1528-1537.

17. Salt, J.; Cuenca, A.; Palau, F.; Dormido, S. A Multirate Control Strategy to the Slow Sensors Problem: An Interactive Simulation Tool for Controller Assisted Design. *Sensors* **2014**, vol.14, n.3, pp.4086-4110, doi:10.3390/s140304086.

18. Sala, A.; Cuenca, A.; Salt, J. A retunable PID multi-rate controller for a networked control system, *Information Sciences* **2009**, vol.179, pp.2390-2402.

19. Cuenca, A.; Salt, J.; Sala, A.; Pizá, R. A delay-dependent dual-rate PID controller over an Ethernet network. *IEEE Trans. on Industrial Informatics* **2011**, vol.7, n.1, pp.18-29.

20. Cuenca, A.; Salt, J.; Casanova, V.; Pizá, R. An Approach Based on an Adaptive Multi-rate Smith Predictor and Gain Scheduling for a Networked Control System: Implementation over Profibus-DP. *Int. J. of Control, Automation, and Systems* **2010**, vol.8, n.2, pp.473-481.

21. Bolot, J. C. End-to-end packet delay and loss behavior in the Internet. *ACM SIGCOMM Computer Communication Review* **1993**, vol.23, n.4, pp.289-298.

22. Mukherjee, A. On the dynamics and significance of low frequency components of Internet load. Technical Reports (CIS) Dep. of Computer & Information Science. Univ. Pennsylvania, **1992**.

23. Mondié, S.; Lozano, R.; Collado, J. Resetting process-model control for unstable systems with delay. *Proc. of the IEEE Conf. on Decision and Control* **2001**, vol.3, pp.2247-2252.

24. Mondié, S.; García, P.; Lozano, R. P. Resetting Smith predictor for the control of unstable systems with delay. *Proc. of the IFAC 15th Triennial World Congress* **2002**.

25. Tipsuwan, Y.; Chow, M-Y. Gain Scheduler Middleware: A Methodology to Enable Existing Controllers for Networked Control and Teleoperation - Part I: Networked Control. *IEEE Trans. on Industrial Electronics*, **2004**, vol.51, n.6, pp.1218-1227.

26. Khargonekar, P.; Poolla, K.; Tannenbaum, A. Robust control of linear time-invariant plants using periodic compensation. *IEEE Transactions on Automatic Control*, **1985**, vol.30, n.11, pp. 1088-1096.

27. Boyd, S; Ghaoui, L. E.; Feron, E.; Balakrishnan, V. *Linear Matrix Inequalities in System and Control Theory*. **1994**, Philadelphia, PA: University City Science Center, Soc. Ind. Math.

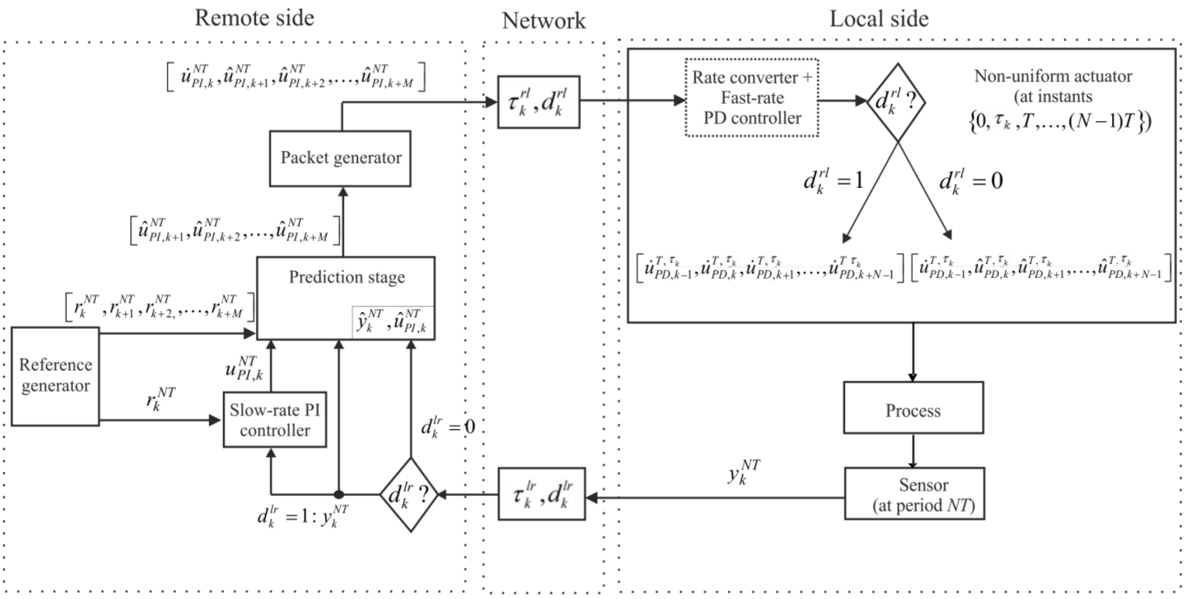
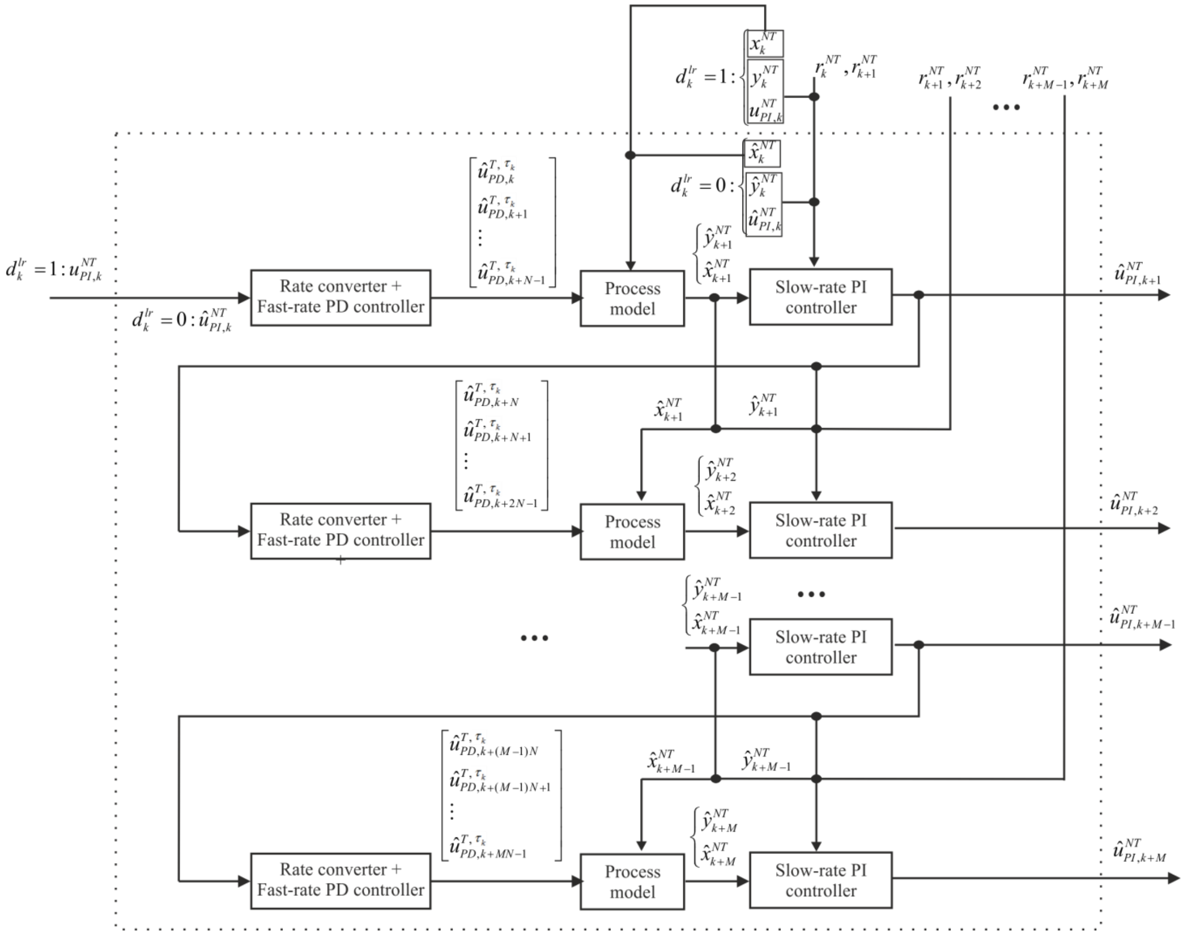
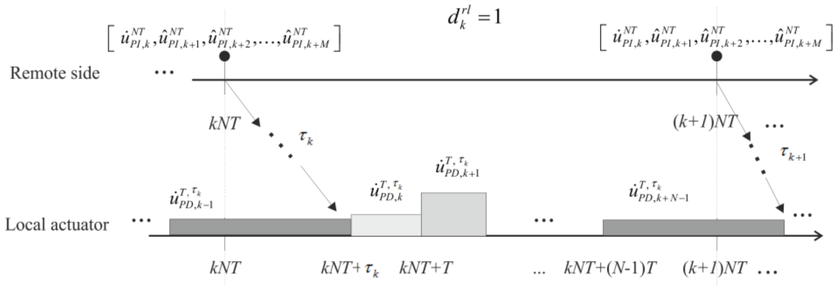

Remote side
Network
Local side
$[\dot{u}^{NT}_{PI,k}, \hat{u}^{NT}_{PI,k+1}, \hat{u}^{NT}_{PI,k+2}, \ldots, \hat{u}^{NT}_{PI,k+M}]$
τ^{rl}_k, d^{rl}_k
Packet generator
$[\hat{u}^{NT}_{PI,k+1}, \hat{u}^{NT}_{PI,k+2}, \ldots, \hat{u}^{NT}_{PI,k+M}]$
Prediction stage
$\hat{y}^{NT}_k, \hat{u}^{NT}_{PI,k}$
$[r^{NT}_k, r^{NT}_{k+1}, r^{NT}_{k+2}, \ldots, r^{NT}_{k+M}]$
Reference generator
$u^{NT}_{PI,k}$
r^{NT}_k
Slow-rate PI controller
$d^{lr}_k = 0$
$d^{lr}_k ?$
$d^{lr}_k = 1 : y^{NT}_k$
τ^{lr}_k, d^{lr}_k
Rate converter + Fast-rate PD controller
$d^{rl}_k ?$
Non-uniform actuator (at instants $\{0, \tau_k, T, \ldots, (N-1)T\}$)
$d^{rl}_k = 1$
$d^{rl}_k = 0$
$[\dot{u}^{T,\tau_k}_{PD,k-1}, \dot{u}^{T,\tau_k}_{PD,k}, \dot{u}^{T,\tau_k}_{PD,k+1}, \ldots, \dot{u}^{T\,\tau_k}_{PD,k+N-1}]$
$[\dot{u}^{T,\tau_k}_{PD,k-1}, \hat{u}^{T,\tau_k}_{PD,k}, \hat{u}^{T,\tau_k}_{PD,k+1}, \ldots, \hat{u}^{T,\tau_k}_{PD,k+N-1}]$
Process
y^{NT}_k
Sensor (at period NT)

Figure 1. NCS scenario.



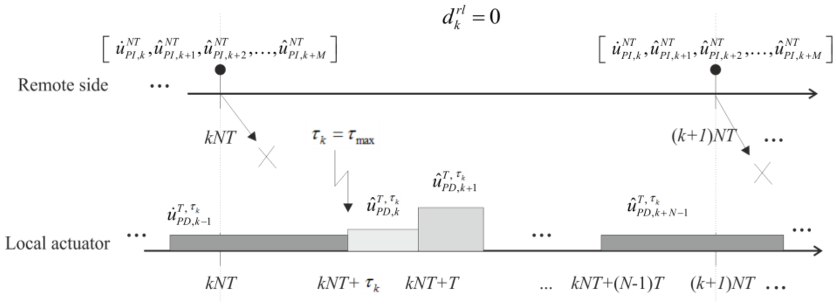
$d_k^{lr}=1: u_{PI,k}^{NT}$
$d_k^{lr}=0: \hat{u}_{PI,k}^{NT}$
$d_k^{lr}=1: \{x_k^{NT}, y_k^{NT}, u_{PI,k}^{NT}\}$
$d_k^{lr}=0: \{\hat{x}_k^{NT}, \hat{y}_k^{NT}, \hat{u}_{PI,k}^{NT}\}$
r_k^{NT}, r_{k+1}^{NT}
$r_{k+1}^{NT}, r_{k+2}^{NT}$
$r_{k+M-1}^{NT}, r_{k+M}^{NT}$
$\hat{u}_{PD,k}^{T,\tau_k}$, $\hat{u}_{PD,k+1}^{T,\tau_k}$, $\vdots$, $\hat{u}_{PD,k+N-1}^{T,\tau_k}$
Rate converter + Fast-rate PD controller
Process model
Slow-rate PI controller
$\hat{y}_{k+1}^{NT}$, $\hat{x}_{k+1}^{NT}$
$\hat{u}_{PI,k+1}^{NT}$
$\hat{u}_{PD,k+N}^{T,\tau_k}$, $\hat{u}_{PD,k+N+1}^{T,\tau_k}$, $\vdots$, $\hat{u}_{PD,k+2N-1}^{T,\tau_k}$
$\hat{x}_{k+1}^{NT}$
$\hat{y}_{k+1}^{NT}$
$\hat{y}_{k+2}^{NT}$, $\hat{x}_{k+2}^{NT}$
$\hat{u}_{PI,k+2}^{NT}$
$\hat{y}_{k+M-1}^{NT}$, $\hat{x}_{k+M-1}^{NT}$
$\hat{u}_{PI,k+M-1}^{NT}$
$\hat{u}_{PD,k+(M-1)N}^{T,\tau_k}$, $\hat{u}_{PD,k+(M-1)N+1}^{T,\tau_k}$, $\vdots$, $\hat{u}_{PD,k+MN-1}^{T,\tau_k}$
$\hat{x}_{k+M-1}^{NT}$
$\hat{y}_{k+M-1}^{NT}$
$\hat{y}_{k+M}^{NT}$, $\hat{x}_{k+M}^{NT}$
$\hat{u}_{PI,k+M}^{NT}$

Figure 2. Prediction stage in detail.



$d_k^{rl}=1$
$[\hat{u}_{PI,k}^{NT}, \hat{u}_{PI,k+1}^{NT}, \hat{u}_{PI,k+2}^{NT}, \dots, \hat{u}_{PI,k+M}^{NT}]$
Remote side
Local actuator
kNT
τ_k
$(k+1)NT$
τ_{k+1}
$\hat{u}_{PD,k-1}^{T,\tau_k}$
$\hat{u}_{PD,k}^{T,\tau_k}$
$\hat{u}_{PD,k+1}^{T,\tau_k}$
$\hat{u}_{PD,k+N-1}^{T,\tau_k}$
$kNT+\tau_k$
$kNT+T$
$kNT+(N-1)T$

Figure 3. No packet dropout ($d_k^{rl}=1$).



$d_k^{rl}=0$
$[\hat{u}_{PI,k}^{NT}, \hat{u}_{PI,k+1}^{NT}, \hat{u}_{PI,k+2}^{NT}, \dots, \hat{u}_{PI,k+M}^{NT}]$
Remote side
Local actuator
kNT
$\tau_k=\tau_{\max}$
$(k+1)NT$
$\hat{u}_{PD,k-1}^{T,\tau_k}$
$\hat{u}_{PD,k}^{T,\tau_k}$
$\hat{u}_{PD,k+1}^{T,\tau_k}$
$\hat{u}_{PD,k+N-1}^{T,\tau_k}$
$kNT+\tau_k$
$kNT+T$
$kNT+(N-1)T$

Figure 4. Packet dropout ($d_k^{rl}=0$).

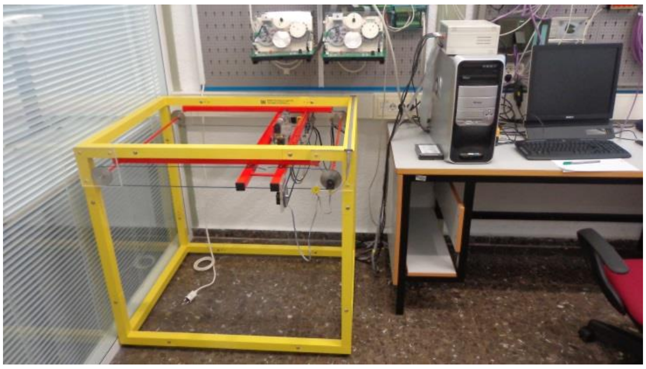


Figure 5. Cartesian robot (3D CRANE module).

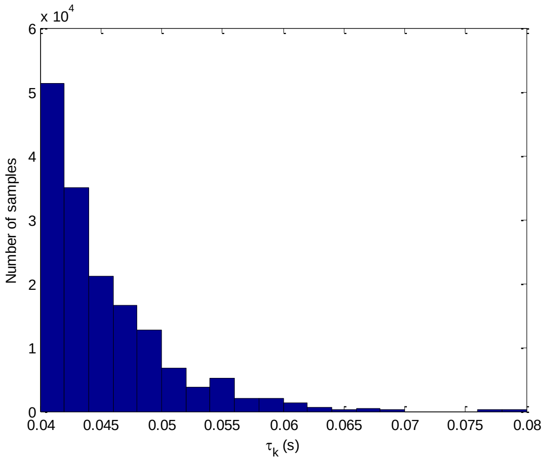


Figure 6. Delay histogram.

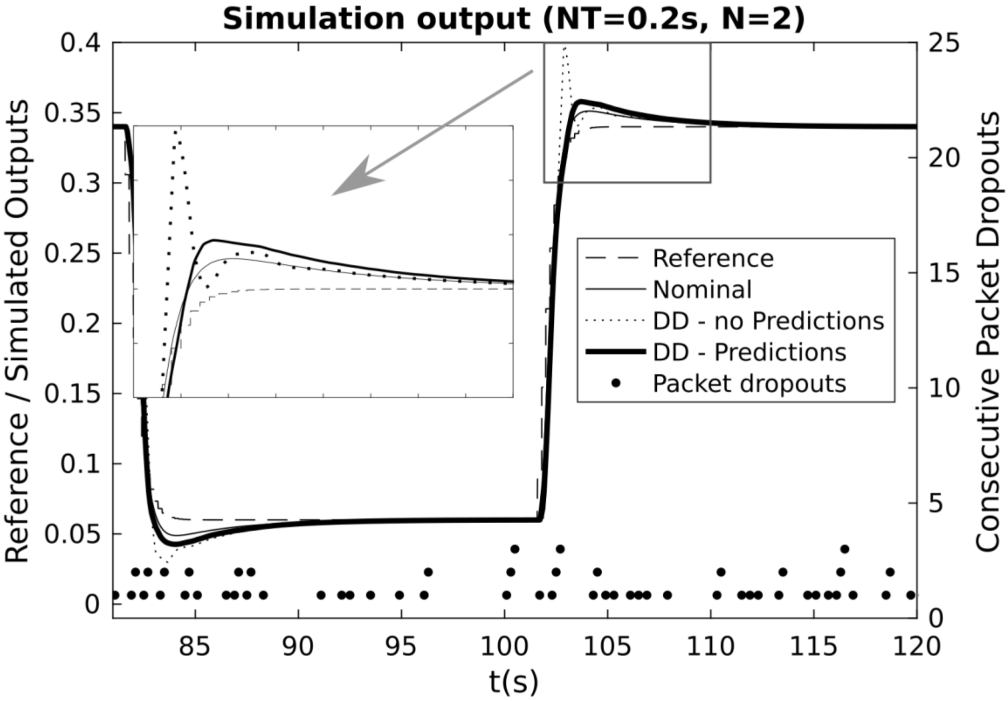


Figure 7. Simulation results: nominal vs delay-dependent with packet dropouts and no prediction vs delay-dependent with packet dropouts with prediction.

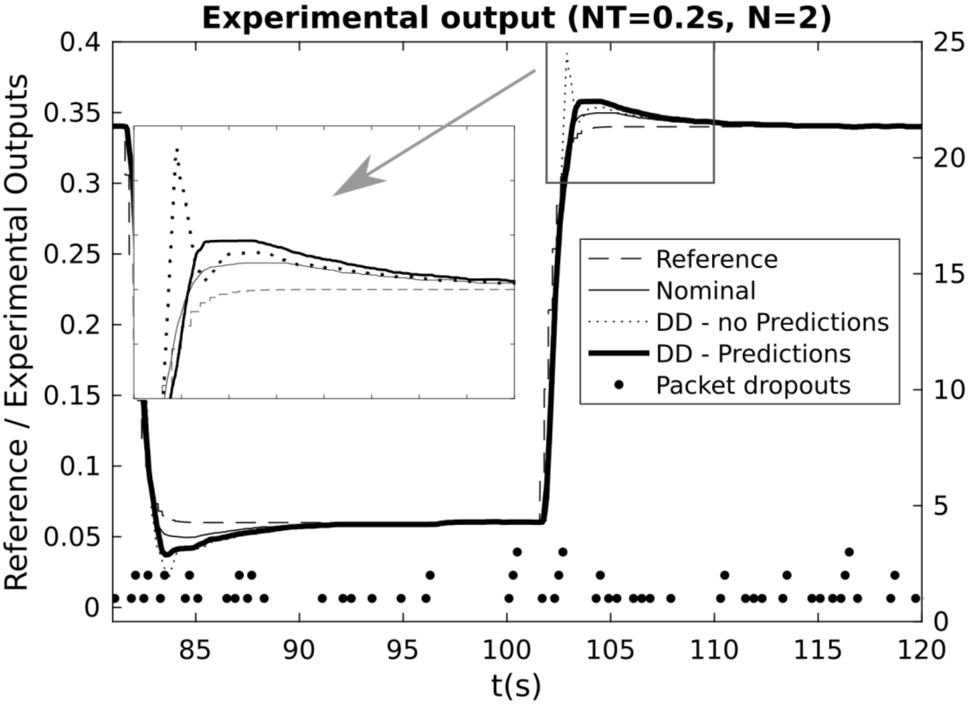


Figure 8. Experimental results: nominal vs delay-dependent with packet dropouts and no prediction vs delay-dependent with packet dropouts with prediction.

Output	E_Y	$J_1(\%)$	O_Y	$J_2(\%)$
Y_{DD-NP}	291.19	0	0.047	0
Y_{DD-P}	168.64	42.09	0.007	85.64
Y_{Nom}	0	100	0	100

Table 1. Cost indexes J_1 and J_2 in simulation.

Output	E	$J_1(\%)$	O	$J_2(\%)$
Y_{DD-NP}	290.55	0	0.042	0
Y_{DD-P}	207.53	28.58	0.012	70.80
Y_{Nom}	0	100	0	100

Table 2. Cost indexes J_1 and J_2 in experimentation.