# Packet-Based Dynamic Control of a Furuta Pendulum over Ethernet

T. Fabbri, D. Fenucci, S. Falasca, M. Gamba, A. Bicchi

***Abstract*— This paper presents experimental results of the application of *Packet-Based Control* approach with dynamic controller on a real plant: the Furuta Pendulum. Despite its easiness of realization, Furuta Pendulum presents some features useful for our purposes, such as pretty non-linear dynamics, unstable equilibrium point and heavy dynamical inaccuracies. The network communication channel has been implemented using an Ethernet network. Packet-Based controller has been tested and compared with a classic local controller for different time-varying actuation delays. Results obtained corroborate the validity of the proposed architecture and highlight the robustness of this approach in the presence of actuation delays.**

***Index Terms*— Distributed systems, Non-linear systems, Robust control, Networked control system**

## I. Introduction

In the past decade, Networked Control Systems (NCSs) have experienced a dramatic growth due to their potential applications in different fields, i.e. manufacturing, intelligent vehicles, power-line control etc.

A NCS is a system in which components such as sensors, actuators and controllers are spatially distributed and connected via a shared communication channel, which can be either a control-oriented infrastructure or a network which is not designed for control purposes like Ethernet. The latter one has many advantages such as small implementation and maintenance costs, scalability and large diffusion; on the other hand, it introduces some constraints which can not be ignored in the control design.

Little can be found in literature concerning experimental results in this field: in [7] experimental results, concerning a dryer controlled via wifi, were provided to corroborate the theoretical achievements. With very few exceptions (e.g. [8], which controls a DC motor over the internet), however, it is difficult to find experiments concerning new theoretical progresses. It is the authors' opinion that the field of control theory is now mature enough to start experimenting with more complex dynamical plants.

The research leading to these results has received funding from the European Union Seventh Framework Programme [FP7/2007-2013] under grant agreement n257462 HYCON2 Network of excellence

T. Fabbri is with University of Pisa, Via Diotisalvi 2, 56126 Pisa, Italy. atfabbri@gmail.com

D. Fenucci is with University of Pisa, Via Diotisalvi 2, 56126 Pisa, Italy. d.fenucci@gmail.com

S. Falasca is with Interdipartmental Research Center "E. Piaggio",Via Diotisalvi 2, 56126 Pisa, Italy. stefano.falasca@centropiaggio.unipi.it

M. Gamba is with University of Pisa, Via Diotisalvi 2, 56126 Pisa, Italy. massimiliano.gamba@gmail.com

A. Bicchi is with Interdipartmental Research Center "E. Piaggio",Via Diotisalvi 2, 56126 Pisa, Italy and IIT - Istituto Italiano di Tecnologia, Genova, Italy. bicchi@centropiaggio.unipi.it

In this paper we show the networked stabilisation (via Ethernet) of a Furuta Pendulum: it is a highly non-linear plant having an unstable equilibrium and very fast dynamics. Moreover it is easy and relatively inexpensive to build. For all these reasons it represents a good benchmark for experiments on networked control approaches.

A promising approach to the problem of stabilising a NCS, presented in [2] and [5], is the *Packet-Based Control* (PBC) which uses an a-priori known static-feedback control law to stabilise the system despite the negative effects due to the communication network in use. The previous strategy has been extended in [3] to the case of dynamic controllers. Such a control architecture is well suited for controlling medium to large plants and provides easy means for both rapid-prototyping of the control and for its deployment.

The realisation of the test-bed made it possible to get some insights on the advantages and disadvantages which are encountered when designing a networked or a classical control architecture.

The experiments proposed in this paper have two goals: assessing the real-world applicability of the approach proposed in [3] and evaluating how the network-induced delays influence the closed-loop system behaviour. Moreover, in this paper a discussion is carried out about the different problems we encountered in implementing both the networked control scheme and the local one. Such discussion is aimed at helping the interested reader in choosing which way to go.

In the following, we provide both a non-linear and a linearised model of the real plant; afterwards a control law able to both swing-up and stabilise the pendulum in the upright position is designed. We remark that, since the purpose of this paper is to illustrate how a real system can be stabilised by using a networked architecture in spite of massive network delays, the used control law is very simple and is by no means being proposed as an alternative to existing, more complex control laws. Experimental results are presented in form of performance comparisons between a classic local controller and the proposed network approach for different time-varying delays.

The results obtained through experiments have been collected in a video which compares the behavior of the locally controlled system with the network controlled system. Moreover, the video includes an animation showing how control data is exchanged among network nodes and used both on the plant side and controller side (which is an animated version of Figure 1). Finally, in the video the behavior of the system under external perturbation is shown. This video is available at IEEE Xplore.

The paper is organised as follows. Section II shows the

main features of the Packet-Based Control approach. Then, in section III dynamical equations of the model of Furuta Pendulum are developed; from these, a linear model of the system is obtained and then the real plant is briefly described. Next, the proposed control law is illustrated in section IV. Finally, section V highlights the major experimental results obtained from the simulations and section VI contains considerations about the implementation of the control scheme.

## II. Packet-Based Control Strategy

This section recalls the main assumptions needed by the Packet-Based Dynamic control strategy with an emphasis on providing details about how—and to which extent—they are met within the experimental setup.

Packet-Based Control is a control approach which aims at compensating for effects due to the use of a packet-switching network. When used for control purposes, communication networks introduce several constraints; those considered by PBC are: large time-varying communication delays, variable transfer intervals, non-simultaneous accesses to the network. The framework developed relies on the algorithm proposed in [3], whose basic idea is schematically illustrated in Figure 1.

The control technique used here for tackling networked control issues belongs to the class of emulation-based approaches. Being emulation-based means that the control law is designed without taking into account the network effects, while the framework provides *automatic* means of adapting the control law to the network infrastructure. In this spirit we will assume that for the system at hand, a stabilising controller has been designed (see assumption 1):

We address the stabilisation of a nonlinear continuous-time system of the form

$$\begin{aligned} \dot{x}_p &= f_p(x_p, u) && (1)\\ y &= g_p(x_p), && (2) \end{aligned}$$

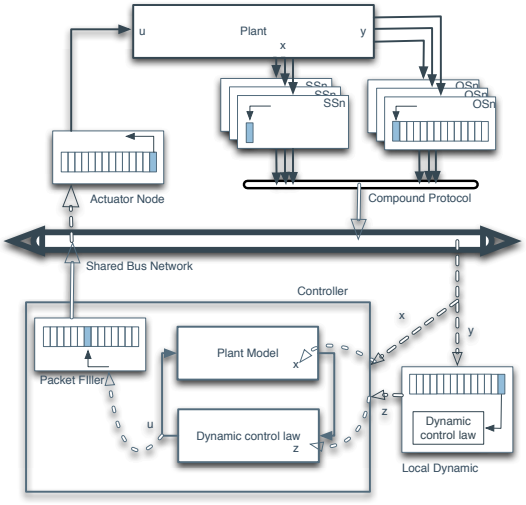



Fig. 1. Schematic illustration of the used control architecture

where $x_p : \mathbb{R}_{\geq 0} \to \mathbb{R}^{n_p}$ is the plant state, $y : \mathbb{R}_{\geq 0} \to \mathbb{R}^{n_y}$ is the output, $u : \mathbb{R}_{\geq 0} \to \mathbb{R}^{n_u}$ represents the control input, and $f_p : \mathbb{R}^{n_p} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_p}$ and $g_p : \mathbb{R}^{n_p} \to \mathbb{R}^{n_y}$ denote locally Lipschitz functions. For this system, we assume that a nominal dynamic feedback controller of the form

$$\begin{aligned} \dot{x}_c &= f_c(x_c, y) && (3)\\ u &= g_c(x_c, x_p, y) && (4) \end{aligned}$$

is available. Here $x_c : \mathbb{R}_{\geq 0} \to \mathbb{R}^{n_c}$ is the controller state, and $f_c : \mathbb{R}^{n_c} \times \mathbb{R}^{n_y} \to \mathbb{R}^{n_c}$ and $g_c : \mathbb{R}^{n_c} \times \mathbb{R}^{n_p} \times \mathbb{R}^{n_y} \to \mathbb{R}^{n_u}$ denote locally Lipschitz functions. Letting $x(t) \triangleq (x_p^T(t), x_c^T(t))^T \in \mathbb{R}^{n_p+n_x} = \mathbb{R}^n$ and

$$\begin{aligned} f(x,u) &\triangleq \begin{pmatrix} f_p(x_p, u) \\ f_c(x_c, g_p(x_p)) \end{pmatrix} \\ g(x) &\triangleq g_c(x_c, x_p, g_p(x_p)), \end{aligned}$$

the closed-loop system (1)-(4) in the absence of network effects simply reads

$$\begin{aligned} \dot{x} &= f(x,u) && (5)\\ u &= g(x). && (6) \end{aligned}$$

*Assumption 1 (Nominal GES):* The origin of the system (1)-(2) in closed-loop with (3)-(4) is globally exponentially stable (GES), i.e. there exists a differentiable function $V : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ and constants $\underline{\alpha}, \overline{\alpha}, \alpha, d > 0$ such that the following conditions hold for all $x \in \mathbb{R}^n$

$$\begin{aligned} \underline{\alpha} \left\|x\right\|^2 \leq V(x) \leq \overline{\alpha} \left\|x\right\|^2 \\ \frac{\partial V}{\partial x}(x) f(x, g(x)) \leq -\alpha \left\|x\right\|^2 \\ \left\| \frac{\partial V}{\partial x}(x) \right\| \leq d \left\|x\right\| . \end{aligned}$$

Moreover, a technical assumption is needed regarding the closed loop system.

*Assumption 2: (Local Lipschitz)* Given some constants $R_x, R_u > 0$, there exist some constants $\lambda_f, \lambda_\kappa > 0$ and all $u_1, u_2 \in B_{R_u}$, the following inequalities hold

$$\left\|f(x_1, u_1) - f(x_2, u_2)\right\| \leq \lambda_f \left(\left\|x_1 - x_2\right\| + \left\|u_1 - u_2\right\|\right) \quad (7)$$

$$\left\|g(x_1) - g(x_2)\right\| \leq \lambda_\kappa \left\|x_1 - x_2\right\| . \quad (8)$$

In the spirit of the emulation-based approach, we only require for the control law to be able to stabilise the plant when used in a classical (non networked) feedback loop. In the experimental setup, we will see that for the designed control law a Lyapunov function guaranteeing the closed loop system to be stable will not be derived; instead, a control law will be used which is known a priori to stabilise the system. This is, in the author's opinion, very important, in that it shows that the whole Packet-Based control framework can be applied to *convert* an existing control loop to a networked one; and this is exactly what being emulation-based is intended to allow for in practical applications.

In the used control setup every communication between the plant (i.e. sensors and actuators) and the controlling computer is carried out through a network. The following

assumption describes the constraints we require for the communication allowed by the network.

*Assumption 3: (Network)* The communication network satisfies the following properties:

i) **(MATI)** There exist two constants $\tau^m, \tau^c \in \mathbb{R}_{\geq 0}$ such that $\tau^m_{i+1} - \tau^m_i \leq \tau^m$ and $\tau^c_{i+1} - \tau^c_i \leq \tau^c$, $\forall i \in \mathbb{N}$;

ii) **(mTI)** There exist constants $\varepsilon^m, \varepsilon^c \in \mathbb{R}_{\geq 0}$ such that $\varepsilon^m \leq \tau^m_{i+1} - \tau^m_i$ and $\varepsilon^c \leq \tau^c_{i+1} - \tau^c_i$ $\forall i \in \mathbb{N}$.

iii) **(MAD)** There exist two constants $T^m, T^c \in \mathbb{R}_{\geq 0}$ such that $T^m_i \leq T^m$ and $T^c_i \leq T^c$, $\forall i \in \mathbb{N}$;

Properties *i)* and *ii)* state that the inter-sending time is lower and upper bounded both on the control side and on the measurement side of the network. In network control theory, the upper bound in *i)* is referred to as *Maximum Allowable Transfer Interval* (MATI). Property *iii)* states that the delays are bounded.

In the experimental setup an Ethernet link is used for communicating between the plant and the controller nodes. Of course, the Ethernet protocol does not guarantee for our assumptions to be verified. In particular, there is no guarantee that a packet incurs a limited delay. In practice, however, this is not an issue, as shown in the network-in-the-loop experiment carried out in [3].

In [3], it is also tacitly assumed that network nodes are synchronised with each other. This characteristic of network nodes will allow the controller to rely on timestamps contained into packets having different sources. As we will see, timestamping is also used for assigning a time-reference to control packets.

In the experimental setup a simple master-slave synchronisation protocol has been implemented, which is sufficient to guarantee the synchronisation.

In order to implement the proposed control strategy, a mathematical model for the plant needs to be available at the controlling computer site. Of course, this model is not required to be perfect. The presented control technique explicitly tackles the robustness problem and allows for a sector-bounded uncertainty to be in place. More precisely, we assume the following:

*Assumption 4: (Sector-Bounded Model Inaccuracy)* Given $R_x, R_u > 0$, there exists a constant $\lambda_{f\hat{f}} \geq 0$ such that, for all $x \in B_{R_x}$ and all $u \in B_{R_u}$,

$$\left\| f(x,u) - \hat{f}(x,u) \right\| \leq \lambda_{f\hat{f}} \left( \|x\| + \|u\| \right). \tag{9}$$

A very important role is played by the network protocol which we now briefly introduce prior to stating the related assumption. The network is a shared channel of communication between the various nodes. Therefore a rule describing the possibility of a node to gain access to the network is needed. Many rules are possible, two examples are the Round Robin and the Mef-Tod protocols. In the Round Robin policy the nodes are allowed to communicate in a cyclic predeterminated order. The Mef-Tod, Maximum error first - Try once discard, consists of letting a node to have access to the network if the error of that sensed variable is the greatest. Once a value is transmitted, some components of the error are set to zero. If a data packets fails to gain access to the network, it is discarded and a new value will be sent in the next access.

Protocols can be viewed in another way i.e. by means of describing the induced change in the information contained in the node that receives an information when a communication is successful. To be more clear, being the protocol the rule the nodes obey in order to communicate through the shared network, it can be described by the update that the packet received by the controller node causes. In the controller node a vector containing the value of the state variable of the plant is maintained. Let $x_p$ and $\hat{x}_p$ be the state of the plant and its estimate. Furthermore let $e_p$ be defined as $e_p \triangleq \hat{x}_p - x_p$. As a packet, containing a measure referred to time $\bar{t}$, reaches the Controller node from the SSn, the error $e_p(\bar{t}) = \hat{x}_p(\bar{t}) - x_p(\bar{t})$ decreases. In fact a piece of information regarding the state of the plant has been received. Typically, but not necessarily, if the number of SSn are $\ell$ the error vector can be partitioned as follows $e = [e^T_{p_1} e^T_{p_2} \dots e^T_{p_\ell}]^T$. The change in the error induced by a reception of a packet containing measures on the state of the plant can be represented as [1]:

$$e_p\left(\tau^{m\ +}_i\right) = h_p\left(i, e_p(\tau^m_i)\right). \tag{10}$$

Although the latter equation is only a map between the error at time $\tau^m_i$ and the error at time $\tau^{m\ +}_i$, it is useful to define an auxiliary discrete-time system of the form:

$$e_p(i+1) = h_p\left(i, e_p(i)\right). \tag{11}$$

Equation (11) is referred to as the *discrete-time system induced by the protocol* or, with some abuse of terms, simply *the protocol*. The core role of the protocol is to ensure that at each transmission some positive definite function of the error $e_p$ is decreased. Consequently, the protocol can be viewed as a tool that allows for reducing the error on the state of the plant. More about the protocols can be found in [7] and [6].

In [3] the protocol is assumed to induce an exponential decrease of the error $e_p$ when the inter-sample dynamics are neglected. More precisely, the following is assumed.

*Assumption 5 (UGES Protocol):* The protocol (10) is uniformly globally exponentially stable and admits an associated Lyapunov function with bounded gradient. More precisely, there exists a function $W_p : \mathbb{N} \times \mathbb{R}^{n_p} \to \mathbb{R}_{\geq 0}$ locally Lipschitz in its second argument and there exist constants $\underline{a}_p, \overline{a}_p, c_p > 0$ and $\rho_p \in [0,1)$ such that for the *discrete-time system induced by the protocol*, i.e. (11), hold the following:

$$\begin{aligned} \underline{a}_p \|\xi\| \leq W_p(i,\xi) \leq \overline{a}_p \|\xi\| \\ W_p\left(i+1, h_p(i,\xi)\right) \leq \rho_p W_p(i,\xi) \end{aligned} \tag{12}$$

for all $\xi \in \mathbb{R}^{n_p}$ and all $i \in \mathbb{N}$. Moreover:

$$\left\| \frac{\partial W_p}{\partial \xi}(i,\xi) \right\| \leq c_p \tag{13}$$

for almost all $\xi \in \mathbb{R}^{n_p}$ and all $i \in \mathbb{N}$.

[1] We tacitly introduced the notation $f(t^+) := \lim_{s \to t, s > t} f(s)$

We remark that the conditions supposed to hold for the *discrete-time system induced by the protocol* do not make any reference to the evolution of the plant nor to the evolution of its model. These conditions captures intrinsic properties of the protocol itself.

Of course in the experimental setup measures are taken by means of digital sensors and are therefore affected by quantisation errors. Strictly speaking, the sending protocol we employ is not UGES, in that it does not correct for quantisation errors.

In the plant side, there are sensor and actuator nodes which are devices that in addition to provide the connectivity to the network, are equipped with transducers depending on the system under consideration. The sensor nodes collect a fixed number of samples from their own transducers and the data set is stored in a local buffer. The whole buffer content is time-stamped and encoded into packets, which are sent to the controller when the network is available. Data which has been already sent is discarded from the buffer. In the controller side, when the data is received, the controller initialises an internal model of the Furuta Pendulum and based on the simulation results, a fixed number of commands to be used in the future is computed. These commands are indexed with respect to time using a timestamp. All information is encapsulated in a packet and sent to the actuator node. In the plant side, the actuator node receives the generated command horizon and it starts running its content by selecting the right control to be used at a given time; in particular, the actuator node compares the timestamp of the last packet it received with its internal clock and moves within the control sequence up to the starting point; moreover, if a new a control sequence is received by the actuator node, the old one is discarded.

## III. System Description

In this section dynamical equations for the Furuta Pendulum system are derived; afterwards a linear model to be used for the stabilisation is provided and finally the real plant is described.

### A. Modelling of pendulum

The Furuta Pendulum includes a horizontal arm coupled to a motor shaft; the pendulum rod, which is free to rotate, is hinged to the horizontal arm. A schematic illustration of Furuta Pendulum is shown in Figure 2. By applying Langrange formulation the dynamics of any multi-link robot can be represented by:

$$B(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + G(\boldsymbol{q}) = \boldsymbol{\tau} \tag{14}$$

where $\boldsymbol{q}$ is the vector of the Lagrangian coordinates, $B(\boldsymbol{q})$ represents the inertia matrix, $C(\boldsymbol{q},\dot{\boldsymbol{q}})$ is the Coriolis, centrifugal and friction matrix, $G(\boldsymbol{q})$ is the gravity matrix and $\boldsymbol{\tau}$ includes the control torques applied to each joint.

In this case, the vector of configuration variables is represented by $\boldsymbol{q} = [q_1, q_2]^T$, where $q_1$ is the angular position of the arm and $q_2$ is the angular position of the pendulum;

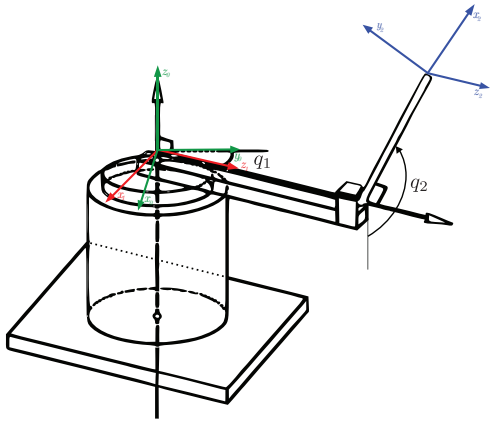



Fig. 2. Schematic representation of Furuta Pendulum

moreover, the considered system is under-actuated, in particular is effectively actuated only the arm joint. The dynamical equation (14) then becomes:

$$\begin{bmatrix} \pi_1 + \pi_2 \sin^2 q_2 + \pi_3 \cos^2 q_2 & \pi_4 \cos q_2 \\ \pi_4 \cos q_2 & \pi_7 \end{bmatrix} \ddot{\boldsymbol{q}} + \begin{bmatrix} \pi_6 + \pi_5 \sin(2q_2)\dot{q}_2 & -\pi_4 \sin q_2 \dot{q}_2 + \pi_5 \sin(2q_2)\dot{q}_1 \\ -\pi_5 \sin(2q_2)\dot{q}_1 & \pi_8 \end{bmatrix} \dot{\boldsymbol{q}} + \begin{bmatrix} 0 \\ \pi_9 \sin q_2 \end{bmatrix} = \begin{bmatrix} \tau \\ 0 \end{bmatrix} \tag{15}$$

where the quantities $\pi_i$ represent the dynamic parameters of the system, which are defined, based on the mechanical parameters reported in Table I, as follows:

$$\begin{aligned} \pi_1 &= J_{z_0} + m_1 l_1^2 + m_2 L_1^2 & \pi_2 &= J_{y_2} + m_2 l_2^2 \\ \pi_3 &= J_{x_2} & \pi_4 &= m_2 L_1 l_2 \\ \pi_5 &= \frac{1}{2}\left(J_{y_2} - J_{x_2} + m_2 l_2^2\right) & \pi_6 &= c_1 \\ \pi_7 &= J_{z_2} + m_2 l_2^2 & \pi_8 &= c_2 \\ \pi_9 &= m_2 l_2 g \end{aligned}$$

where $g$ is the gravity.

TABLE I
Parameters of Furuta pendulum

| Physical quantity | Symbol | Units | |
|---|---|---|---|
| Arm mass | $m_1$ | $200 \times 10^{-3}$ | [kg] |
| Pendulum mass | $m_2$ | $72 \times 10^{-3}$ | [kg] |
| Arm length | $L_1$ | $224 \times 10^{-3}$ | [m] |
| Arm COM | $l_1$ | $144 \times 10^{-3}$ | [m] |
| Pendulum COM | $l_2$ | $106 \times 10^{-3}$ | [m] |
| Arm $z_0$ inertia | $J_{z_0}$ | $0.9 \times 10^{-3}$ | [kg m$^2$] |
| Pendulum $x_2$ inertia | $J_{x_2}$ | $1.65 \times 10^{-6}$ | [kg m$^2$] |
| Pendulum $y_2$ inertia | $J_{y_2}$ | $2.7 \times 10^{-4}$ | [kg m$^2$] |
| Pendulum $z_2$ inertia | $J_{z_2}$ | $2.71 \times 10^{-4}$ | [kg m$^2$] |
| Arm friction | $c_1$ | $0.9 \times 10^{-2}$ | [N m s] |
| Pendulum friction | $c_2$ | $2.71 \times 10^{-7}$ | [N m s] |
| Motor torque constant | $K$ | 2.2274 | [N m A$^{-1}$] |
| Motor inductance | $L_a$ | 0.044 | [H] |
| Motor resistance | $R_a$ | 1.9 | [$\Omega$] |

In order to change the upright position from $q_2 = \pi$ to $q_2 = 0$ we provide the following coordinate transformation:

$$\boldsymbol{\theta} = \begin{bmatrix} q_1 \\ q_2 - \pi \end{bmatrix}$$

Using the previous change of variables, a state space model of the system (15) with the unstable equilibrium in the origin, can be represented, choosing as state vector $\boldsymbol{x} = \left[\boldsymbol{\theta}^T, \dot{\boldsymbol{\theta}}^T\right]^T$, by:

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{\boldsymbol{\theta}} \\ -B(\boldsymbol{\theta})^{-1}\left[C(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} + G(\boldsymbol{\theta})\right] \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ B(\boldsymbol{\theta})^{-1} \end{bmatrix} \boldsymbol{\tau} \quad (16)$$

The torque $\tau$ is generated by a DC motor, modelled by the following first order linear dynamic:

$$L_a \dot{\tau} = KV - R_a \tau - K^2 \dot{\theta}_1 \quad (17)$$

where $V$ is the voltage applied to the motor and $K$, $L_a$, $R_a$ are the motor parameters described in Table I.

### B. Linear Model

The linear model of the mechanical plant is obtained by linearising the dynamical equation in (16) around the upright state and is described by the following equation:

$$\dot{\boldsymbol{x}} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{\pi_4 \pi_9}{d} & \frac{-\pi_6 \pi_7}{d} & \frac{-\pi_4 \pi_8}{d} \\ 0 & \frac{\pi_9(\pi_1+\pi_3)}{d} & \frac{-\pi_4 \pi_6}{d} & \frac{-\pi_8(\pi_1+\pi_3)}{d} \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} 0 \\ 0 \\ \frac{\pi_7}{d} \\ \frac{\pi_4}{d} \end{bmatrix} \tau \quad (18)$$

where $d = -\pi_4^2 + \pi_7(\pi_1 + \pi_3)$.

The previous equation has been combined with the equation (17), considering the torque $\tau$ as an additional state and the voltage $V$ as the input, to obtain the linear model of the complete system (i.e. pendulum + motor).

### C. Real plant

The Furuta Pendulum used for experiments is shown in Figure 3. The mechanical system is equipped with two encoders: a very precise one measures the $\theta_1$ angle with $2\pi/176128\,\mathrm{rad}$ resolution, while the second one measures $\theta_2$ with $2\pi/2000\,\mathrm{rad}$ resolution. The horizontal arm is actuated by a DC motor, which has an input voltage range of $-24 \div 24\,\mathrm{V}$. Clearly, the real plant has some physical phenomena which are not modelled, like Coulomb frictions and the play between the arm and the motor due to the presence of a reduction gear.

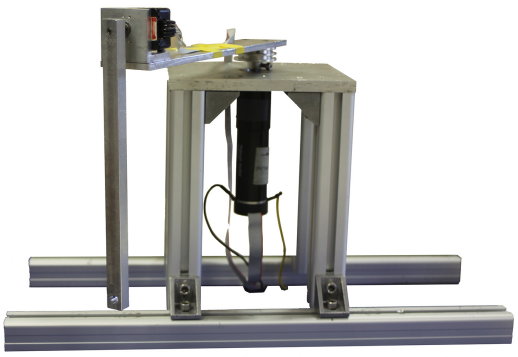


Fig. 3. Furuta Pendulum system

## IV. Control Design

The aim of the control is divided in two distinct steps: the first one is the *Swing-Up* phase, in which the pendulum starts from the downward state and has to reach the upright position; the second one is the *Stabilisation* phase, where the controller must be able to maintain stable the pendulum around this position. The switching from the Swing-Up phase to the other one occurs when the pendulum enters in an angular sector in the neighborhood of the upright position. In this section control laws for both swing-up and stabilisation phases are provided.

### A. Swing-Up control

A strategy for bringing the pendulum to the upright position is shown in [1]. Assume that the pendulum is at rest in the downward position; the basic idea is to apply a constant torque in an arbitrary direction to the arm and reverse it when the velocity of the pendulum becomes zero; through this approach the angle described by the pendulum doubles every time the torque applied to the arm is reversed. Due to the difficulty of estimating accurately the velocity of the pendulum, we have used a simplified version of the previous law, summarised as follows:

- a positive torque when the pendulum has positive velocity and $mod(\theta_2, 2\pi)$ is less than $\pi$;
- a negative torque when the pendulum has negative velocity and $mod(\theta_2, 2\pi)$ is greater than $\pi$;
- a null torque otherwise.

With reference to the equation (17), neglecting the transient of the motor ($\dot{\tau} = 0$), the voltage to be applied to the motor in order to obtain the torque $\bar{\tau}$ as defined previously is equal to:

$$V = \frac{R_a \bar{\tau} + K^2 \dot{\theta}_1}{K}$$

### B. Stabilisation control

The stabilisation law is obtained through a state observer plus a state feedback of the linearised model of the pendulum described by the equations (18), (17). In order to stabilise the system in the upright position, i.e. $\boldsymbol{x} = [0 \quad 2k\pi \quad 0 \quad 0]^T, \tau = 0$ with $k \in \mathbb{Z}$, the Luenberger observer has as inputs, in addition to the angular position of the arm and the controlled voltage, the angular position error due to the difference between the reference, equal to the nearest upright position, and the current angular position of the pendulum (i.e. if the pendulum is located between $-\pi$ e $\pi$ the reference applied is 0).

## V. Experimental Results

The main goal of the carried experiments is to characterise the influence of network-induced delays on the control. In particular, the PBC approach has been tested for different random time-varying delays. The random delays have been

generated according to a Gaussian distribution; in different tests, the mean value of the distribution is changed spanning from $0$ with $0$ variance (no delay is added to the delay induced by the network in use) to $30$ (an average $30\times10^{-3}[\mathrm{s}]$ delay is artificially induced) with $2$ variance. The relative ordering of the packets is forced to be maintained.

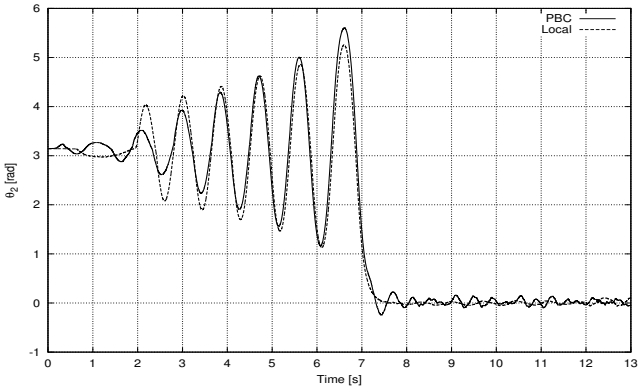
In this section experimental results obtained with various tests on the real plant are illustrated. First, a comparison between the local and the packet-based networked control for different time-varying actuation delays is provided; afterwards, all the results obtained with PCB strategy are summarised in a single graph in order to display the difference in the performance with increasing time delays.

Local control has been realised using a micro-controller with two hardware encoder modules for reading the angular positions of the two link and a PWM module that controls the motor.

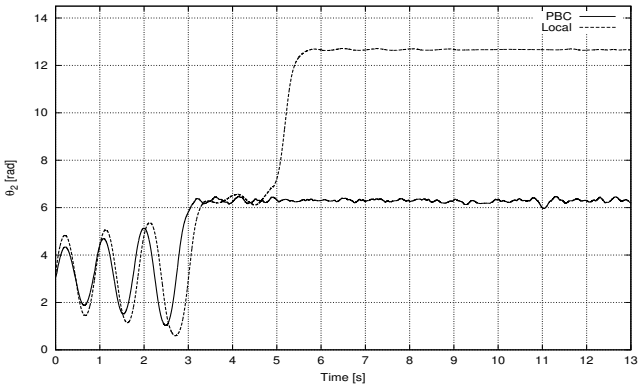
On the other end, for the networked control experiments a software for networked control systems based on the one presented in [4] has been used. Modules for the embedded platform used in physical network nodes have been written for this purpose.

In the PBC test-bed implementation the control network is made of three nodes: the controller, an actuator/sensor node for $\theta_1$ and a sensor node for $\theta_2$. The communication link is an Ethernet network having a star topology. Packets are sent via UDP. The test-bed uses an 88 bytes long payload when sending control packets: each packet contains the control for the next $40\,\mathrm{ms}$ (i.e. 40 times the sampling time used for control). Sensors send 80 bytes with every packet. Sensor data set contains $10$ samples and is sent every $10\,\mathrm{ms}$. Sensor and actuator nodes have been implemented on a micro-controller using the C language, the computer software for the controller is written in C++. In our implementation, the time needed for computing the control law over a time-horizon of $40\,\mathrm{ms}$ is $4\,\mathrm{ms}$.

The results obtained and reported below exhibit time

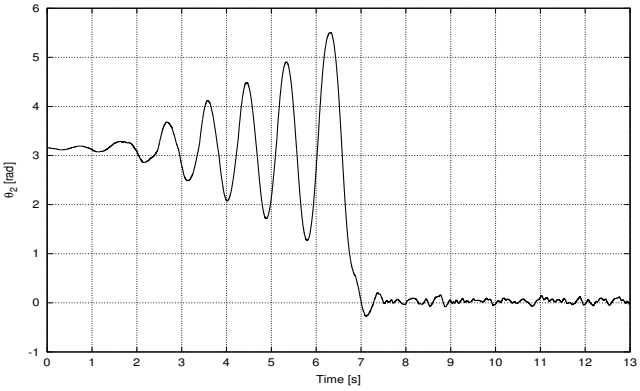



(a) No delay

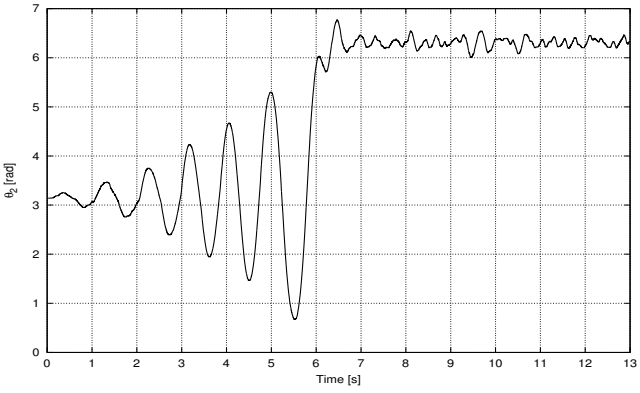



(b) Time delay $\mathcal{N}(5,2)$ ms

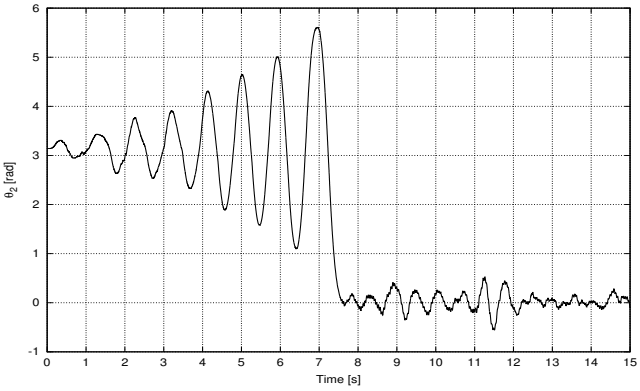
Fig. 4. Comparison between local and packet-based network control




(a) Time delay $\mathcal{N}(10,2)$ ms




(b) Time delay $\mathcal{N}(20,2)$ ms




(c) Time delay $\mathcal{N}(30,2)$ ms

Fig. 5. Packet-based network control with different time-varying delays

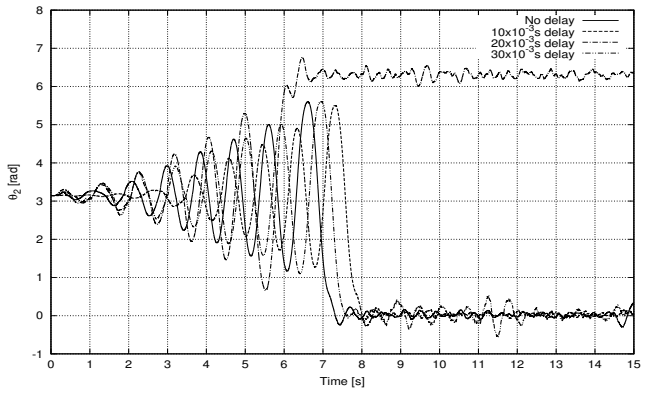



Fig. 6. Packet-based network control: comparison between different time-varying delays

responses of $\theta_2$. Figure 4 shows the difference in behaviour between the local and the packet-based networked controller: in Figure 4(a), where there is no actuation delay, responses are very similar, but as soon as a Gaussian actuation delay (with $5\,\mathrm{ms}$ on average and variance 2) is introduced as in Figure 4(b), the performance of the local controller is degraded. For actuation delays with higher means, the local controller is no longer able to execute the swing-up nor to stabilise the system; for this reason in Figure 5 are shown only the results obtained with PBC for delays on increasing average: it's clear that the more the mean of the actuation delay is high, the more the pendulum fluctuations around the unstable equilibrium are strengthened. Finally, Figure 6 summarises time responses obtained with PBC for various delays.

## VI. Discussion

In order to carry-out the presented experiments, both the networked control scheme and the local one have been implemented. In this section we want to highlight the different problems that arose in designing them.

Even for the simple system we considered, the local control implementation required us to select a micro-controller having two hardware encoder modules. On the other hand, each network node we implemented used a much simpler micro-controller, which has only one hardware encoder module. Of course, this is by no means an issue when controlling a Furuta pendulum, but it might become so for a different system with more sensors/actuators.

The two approaches required radically different efforts for the embedded coding. The local control scheme required to write routines for input/output and for computing the control values. The networked control scheme, on the other hand, required the implementation of input/output and networking routines. The control law computation for the networked version was much simpler in its implementation, being carried out on a full blown PC.

In order to capture the data during the experiments for the local control, an external device had to be used as a data logger. The networked control scheme, on the other hand, already transmits all relevant data over the network, so that it is captured by the controlling computer itself.

That said, it is the authors' opinion that, as expected, the networked control approach is much more convenient when a medium to large plant has to be controlled.

Lastly, the PBC approach and its implementation based on the software tools presented in [4] appears to be a suitable framework for fast-prototyping as well as for the deployment of a control system.

## VII. Conclusions

This paper has presented experimental results which allows for corroborating the real-world applicability of the PBC architecture. The experiments were carried out for a Furuta pendulum controlled during the swing-up phase as well as when keeping the upright position. As for PBC, the controller has been designed without taking into account the effects of the network. The effects of different network delays have been shown. Experience with this control implementation shows that the proposed framework allows for the networked system to mimic the behaviour of a classical local control by compensating even significant network delays. Finally, a comparison between the efforts needed for implementing the networked control system and the local control system has been provided.

## Acknowledgments

Special thanks go to Stefano Melani for his help and support in implementing the embedded software on network nodes.

## References

[1] K. J. Astrom and K. Furuta. Swinging up a pendulum by energy control. In *IFAC 13th World Congress, San Francisco, California, 1996*, 1996.

[2] A. Chaillet and A. Bicchi. Delay compensation in packet-switching networked controlled systems. In *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, pages 3620 –3625, dec. 2008.

[3] S. Falasca, M. Gamba, and A. Bicchi. A strategy for dynamic controller emulation in packet-based networked control systems. *submitted for publication*, 2013.

[4] Stefano Falasca, Christian Belsito, Andrea Quagli, and Antonio Bicchi. A Modular and Layered Cosimulator for Networked Control Systems. *Proc. IEEE Mediterranean Conference on Control*, 2010.

[5] L. Greco, A. Chaillet, and A. Bicchi. Exploiting packet size in uncertain nonlinear networked control systems. *Automatica*, 48:2801–2811, 2012.

[6] D. Nesic and D. Liberzon. A unified framework for design and analysis of networked and quantized control systems. *Automatic Control, IEEE Transactions on*, 54(4):732 –747, april 2009.

[7] G.C. Walsh and Hong Ye. Scheduling of networked control systems. *Control Systems, IEEE*, 21(1):57 –65, feb 2001.

[8] Yun-Bo Zhao, Guo-Ping Liu, and D. Rees. Packet-based deadband control for internet-based networked control systems. *Control Systems Technology, IEEE Transactions on*, 18(5):1057 –1067, sept. 2010.