Cooperative Dynamic MPC for NCSs

Isabel Jurado, Daniel E. Quevedo, Karl H. Johansson and Anders Ahlén

Abstract This work studies cooperative MPC for Networked Control Systems with multiple wireless nodes. Communication between nodes is affected by random packet dropouts. An algorithm is presented to decide at each time instant which nodes will calculate the control input and which will only relay data. The nodes chosen to calculate the control values solve a cooperative MPC by communicating with their neighbors. This algorithm makes the control architecture flexible by adapting it to the possible changes in the network conditions.

1 Introduction

Networked Control Systems (NCSs) are systems in which practical communication links are used to exchange system information and control signals between various components of the system that may be physically distributed. Major advantages of NCSs include low cost, reduced weight and power requirements, simple installation and maintenance, and high reliability. Nonetheless, closing a control loop on a shared communication network introduces additional dynamics and constraints

Isabel Jurado
Departamento de Ingeniería de Sistemas y Automática, Escuela Superior de Ingenieros, Universidad de Sevilla, Spain. e-mail: ijurado@cartuja.us.es

Daniel E. Quevedo
School of Electrical Engineering & Computer Science, The University of Newcastle, NSW 2308, Australia. e-mail: dquevedo@ieee.org

Karl H. Johansson
ACCESS Linnaeus Centre, School of Electrical Engineering, Royal Institute of Technology, Stockholm, Sweden. e-mail: kallej@ee.kth.se

Anders Ahlén
Department of Engineering Sciences, Signals and Systems, Uppsala University, Uppsala, Sweden. e-mail: Anders.Ahlen@signal.uu.se

in the control problem. In addition to being bit-rate limited [13, 6], practical communitacion channels are commonly affected by packet dropouts and time delays, mainly due to transmission errors and waiting times to access the medium; see, e.g., [3]-[16] and the many references therein.

This chapter studies NCS in which the transmissions are affected by random packet dropouts. The network is composed of a certain number of nodes forming a matrix strucure. These nodes follow an algorithm, that decides which node will calculate the control input. This node will solve a cooperative MPC communicating with its neighbors. Each node knows a part of the whole system model and it shares its information with a group of neighbor nodes, so they cooperate in order to exchange their information about the system. At each sampling time, we have a different group of nodes chosen to calculate the control signal. This group of nodes will be chosen depending on the particular network outcomes for that sampling time. The present work extends our recent conference contribution [7] to encompass NCSs with parallel links and the use of cooperative MPC. The idea is motivated by the fact that the link transmission outcomes may change at each sampling instant, so one particular node is not always the best suited to perform the control calculation.

In the network under consideration in this work, the only node that receives the state of the plant without any dropouts is the sensor node, which is located next to the plant. The actuator node is directly connected to the plant input, therefore this data is received without problems. The actuator node is also the only node that provides transmission acknowledgments. We assume as well that there is an array of nodes between the sensor and the actuator nodes, as shown in Fig. 1. Each node in a column sends the information to the three closest nodes in the following column. We will assume that the nodes, except the sensor and the actuator nodes, can communicate with some of its neighbors in the same column, and may thereby cooperate and exchange information. The sensor and actuator nodes can't calculate control values; they can only transmit information. The communication between nodes is limited to a maximum number of iterations, and subject to dropouts.

We are supposing that the model of the plant is divided into a certain number of incomplete subsystems. Each node will know only a part of the model of the system, that is why it has to collaborate with its neighbors, which know the other parts of the system. Therefore, each node will estimate just a part of the state.

The control policy to be used will be a cooperative MPC. Within this context, we present a flexible NCS architecture where the role played by cooperative nodes depends upon transmission outcomes and their acknowledgments. With the algorithm proposed, transmission outcomes and their acknowledgments will determine, at each time instant, whether the control input will be calculated at the actuator node, or closer to the sensor node.

Therefore, the distinguishing feature of this approach is the dynamic architecture of the controlled system, plus the fact that we are using a matrix of nodes which do not know the whole information of the plant model. The algorithm seeks to find the best group of nodes to calculate the control, depending on the network transmission outcomes.

The remainder of this chapter is organized as follows. Section 2 describes the NCS topology of interest. In Section 3.1 we present the control law. Section 3.2 provides the state estimation. Section 3.3 presents the algorithm that chooses the best node to calculate the control action. In Section 4 an example illustrating the control algorithm is presented and in Section 5 conclusions are drawn.

Notation: We write $\mathbb{N}_0$ for $\{0,1,2,\ldots\}$; $\mathbb{R}$ are the real numbers, whereas $\mathbb{R}_{\geq 0} \triangleq [0,\infty)$. The trace of a matrix A is denoted by $\mathbf{tr}(A)$. If a matrix $\mathbf{Q}$ is positive definite, then we write $\mathbf{Q} > 0$. We adopt the convention $\sum_{j=1}^{0} a_j = 0$, for all $a_0, a_1 \in \mathbb{R}$. The expectation of a random variable μ is denoted by $\mathbf{E}\{\mu\}$. A real random variable μ, which is zero-mean Gaussian with covariance Γ is denoted by $\mu \sim \mathcal{N}(0,\Gamma)$.

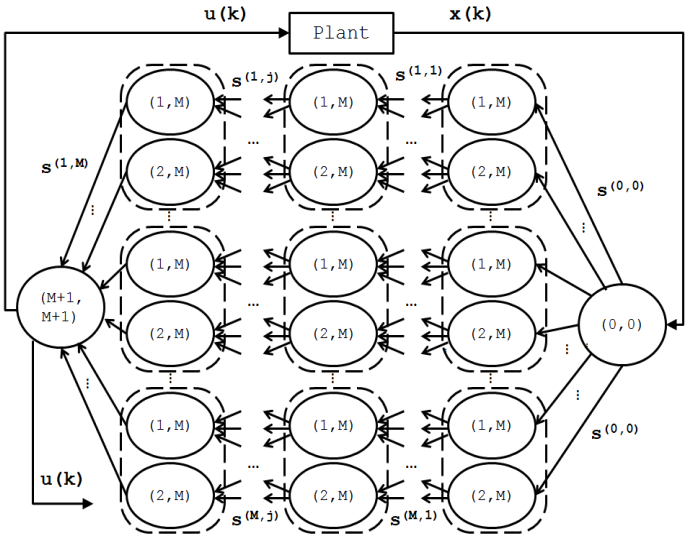

Fig. 1 Control over a graph with dropouts and unreliable acknowledgments of actuator values.

2 Considerations about the System Setup Control Objectives and the Network Constraints

We consider MIMO LTI plant models of the form

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k) + \mathbf{d}(k) \tag{1}$$

where $\mathbf{x}(0) \sim \mathcal{N}(0,\mathbf{P}_0)$, $\mathbf{P}_0 > 0$. In (1), $\mathbf{u}(k) \in \mathbb{R}^{n_u}$ is the plant input, $\mathbf{x}(k) \in \mathbb{R}^{n_x}$ is the state, and $\mathbf{d}(k) \sim \mathcal{N}(0,\mathbf{D})$, $\mathbf{D} > 0$ is driving noise.

The model described in (1) represents the whole plant. But, as foreshadowed in the introduction, individual nodes do not have knowlegde of this whole model. Thus, nodes have to interact with their neighbors to get all the information about the plant. We are considering that, between a certain number of nodes, they have all the information about the plant model. Fig. 1 shows a particular situation in which the whole information is shared by two nodes.

2.1 The composite model

For each node, the *composite model (CM)* [2], is the combination of the decentralized model and all the interaction models. In order to make the problem formulation simpler, we will consider the particular case shown in Fig. 1, in wich the cooperation is done between two nodes in the same column, i.e., we have one interation model. The decentralized state vector in node (i,j), $\mathbf{x}_{(i,j)}$, is augmented with the state from the neighbor node (i^*,j).

Therefore, the augmented state $\mathbf{x}_{(i,j)} = [\mathbf{x}^T_{(i,j)(i,j)}, \mathbf{x}^T_{(i,j)(i^*,j)}]^T$ represents the CM states for the node (i,j), (i^*,j) being the neighbor node interacting, pairwise, with (i,j), which is in the same column and $i^* \in \{i-1, i+1\}$. In this augmented state, $x_{(i,j)(i^*,j)}$ is the influence of the node (i^*,j) on the node (i,j), and $x_{(i,j)(i,j)}$ is the part of the state that take into account just the part of the model that the node (i,j) knows, so it is a decentralized state. In this case, the CM for the node (i,j) is written as

$$\mathbf{x}_{(i,j)}(k+1) = \mathbf{A}_{(i,j)}\mathbf{x}_{(i,j)}(k) + \mathbf{B}_{(i,j)}\mathbf{u}_{(i,j)}(k) + \mathbf{W}_{(i,j)(i^*,j)}\mathbf{u}_{(i^*,j)}(k)$$

where

$$\mathbf{A}_{(i,j)} = \begin{bmatrix} \mathbf{A}_{(i,j)(i,j)} & \\ & \mathbf{A}_{(i,j)(i^*,j)} \end{bmatrix}, \quad \mathbf{B}_{(i,j)} = \begin{bmatrix} \mathbf{B}_{(i,j)(i,j)} \\ \mathbf{0} \end{bmatrix}, \quad \mathbf{W}_{(i,j)(i^*,j)} = \begin{bmatrix} \mathbf{0} \\ \mathbf{B}_{(i,j)(i^*,j)} \end{bmatrix}.$$

Since

$$\mathbf{A}_{(i,j)(i-1,j)} = \mathbf{A}_{(i,j)(i+1,j)} \quad \textit{and} \quad \mathbf{B}_{(i,j)(i-1,j)} = \mathbf{B}_{(i,j)(i+1,j)},$$

$\mathbf{A}_{(i,j)}$, $\mathbf{B}_{(i,j)}$ and $\mathbf{W}_{(i,j)(i^*,j)}$ do not depend on the value of i^*. Therefore, two neighbor cooperating nodes have available the entire information about the system model. Therefore, the node (i,j) has one part of the model, and the neighbor nodes $(i-1,j)$ and $(i+1,j)$ have the other part of the model.

2.2 Control problem

The augmented control signal to calculate is denoted as

$$\mathbf{u}_{(i,j)}^{(i^*,j)}(k) = [\mathbf{u}_{(i,j)}(k)^T, \mathbf{u}_{(i^*,j)}(k)^T]^T, \quad k \in \mathbb{N}_0, \tag{2}$$

and will be calculated with MPC techniques employing pair-wise cooperation among the neighboring nodes. Thus, the control problem solved for this kind of systems is

$$\mathbf{u}_{(i,j)}^{(i^*,j)}(k) = Cooperative \quad MPC\left(\mathbf{x}_{(i,j)}(k), \mathbf{x}_{(i^*,j)}(k)\right), \quad k \in \mathbb{N}_0, \tag{3}$$

where $\mathbf{x}_{(i,j)}(k)$ and $\mathbf{x}_{(i^*,j)}(k)$ represent the CM states for the nodes (i,j) and (i^*,j), respectively.

2.3 Network issues

Sensor and actuator nodes are connected via a wireless network, characterised via a graph having $M \times M + 2$ nodes, see Fig. 1. Control values cannot be calculated by the sensor nor the actuator nodes, they are just used to measure the plant state and apply the control signal, respectively. Therefore, according to Fig. 1, the network has $M \times M$ nodes that could act as the controller. Transmissions are done in a sequential manner as shown in Fig. 2. More precisely, the packet $\mathbf{s}^{(i,j)}(k)$ is transmitted from node (i,j) to its closest neighbors[1] at times $kT + j\tau$, where T is the sampling period of (1) and $\tau \ll T/(M+1)$ refers to the times between transmissions of packets. The plant input $\mathbf{u}(k)$ is applied at time $kT + (M+1)\tau$. We, thus assume that in-network processing is much faster than the plant dynamics (1) and, as in, e.g., [4], neglect delays introduced by the network.

A distinguishing characteristic of the situation at hand is that (due to channel fading) the network introduces packet dropouts. To study the situation, we adopt an analog erasure channel model and introduce the binary success random processes

$$\gamma_{(i,j)}^{(i,j-1)}(k) \in \{0,1\}, \quad k \in \mathbb{N}_0,\ i \in \{0,1,2,\ldots,M+1\}, j \in \{0,1,2,\ldots,M+1\}$$

where $\gamma_{(i,j)}^{(i,j-1)}(k) = 1$ indicates that transmission of the packet $\mathbf{s}^{(i,j-1)}(k)$ from node $(i,j-1)$ to node (i,j) at time $kT + (j+1)\tau$, is successful, i.e., error-free; $\gamma_{(i,j)}^{(i,j-1)}(k) = 0$ refers to a packet-dropout. Throughout this work we assume that the sensor node $(0,0)$ has direct access to plant output measurements. For notational convenience, we write $\gamma^{(0,0)}(k) = 1$, for all $k \in \mathbb{N}_0$.

[1] These are $(i+1,j+1)$, $(i,j+1)$ and $(i-1,j+1)$.

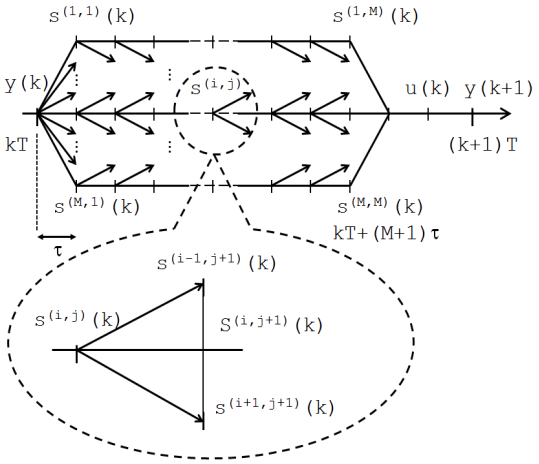

Fig. 2 Transmission Schedule; $t \in \mathbb{R}_{\geq 0}$ is actual time.

To save energy, in our formulation the wireless nodes (i, j), where

$$i, j \in \{0, 1, 2, \ldots, M\},$$

do not provide acknowledgments of receipt of the packets. However, the actuator node, $(M+1, M+1)$, will in general have less stringent energy constraints, so at time $kT + (M+1)\tau$ the control signal is received, at $kT + (M+2)\tau$ this control value is applied and at time $kT + (M+3)\tau$, the actuator broadcasts the control value applied, namely $\mathbf{u}(k) = [\mathbf{u}^A_{(i,j)}(k)^T, \mathbf{u}^A_{(i^*,j)}(k)^T]^T$, back to the wireless nodes (i, j), see Fig. 1. This acknowledgment-like signal is unreliable and affected by dropouts with associated success processes

$$\delta^{(i,j)}(k) \in \{0, 1\}, \quad k \in \mathbb{N}_0,\ i, j \in \{0, 1, 2, \ldots, M\}.$$

More precisely, if $\mathbf{u}(k)$ is successfully received at node (i, j), then we set $\delta^{(i,j)}(k) = 1$; see also [11] and [18] for studies on the importance of acknowledgments in closed loop control. We assume that the actuator node has perfect knowledge of plant inputs, and thus, write $\delta^{(M+1,M+1)}(k) = 1$, $\forall k \in \mathbb{N}_0$.

Due to packet dropouts, plant state measurements are not always available at the actuator node. On the other hand, the sensor node will, in general, not have perfect information of previous plant inputs. This makes the implementation of (3) via estimated state feedback a challenging task. The main purpose of the present work is to decide which nodes of the network (with the exception of the sensor and actuator nodes) should use their local state estimates to implement the control law (3), that is, which node will play the role of the controller and which ones only relay the received information. We foresee that our approach will lead to a dynamic assignment of the role played by the individual network nodes. Which tasks

are carried out by each node at each time instant, will depend upon transmission outcomes, i. e., on $\gamma_{(i,j)}^{(i,j-1)}(k)$ and $\delta^{(i,j)}(k)$.

2.4 Dynamic controller placement

The packets transmitted by each node (i,j) have three fields, namely, state measurements, tentative plant inputs (if available) and the value of the objective function under consideration:

$$\mathbf{s}^{(i,j)}(k) = \left(\mathbf{x}(k), \mathbf{u}_{(\alpha,\beta)}^{(\alpha^*,\beta)}(k), J(k)\right), \alpha \in \{1,\ldots,i\}, \alpha^* \in \{\alpha-1, \alpha+1\}, \beta \in \{1,\ldots,j\}. \quad (4)$$

The plant states $\mathbf{x}(k)$ includes the two components corresponding to the cooperation nodes, that is $\mathbf{x}(k) = [\mathbf{x}_{(i,j)}(k)^T, \mathbf{x}_{(i^*,j)}(k)^T]$, with $i^* \in \{i-1, i+1\}$.

The control signal $\mathbf{u}(k)_{(\alpha,\beta)}^{(\alpha^*,\beta)}$ in (4) with the structure shown in (2), is the plant input which is applied at the plant provided the packet $\mathbf{s}^{(i,j)}(k)$ is delivered at the actuator node. If $\mathbf{s}^{(i,j)}(k)$ is lost, then following Algorithm 2, which will be described in Section 3.3, the plant input will be provided by one of the nodes in subsequent columns, see Fig. 1, which thereby takes on the controller role at time k. For further reference, we will refer to the node which calculates the plant input at time k as

$$c(k) \in \{1,2,\ldots,M\}^2.$$

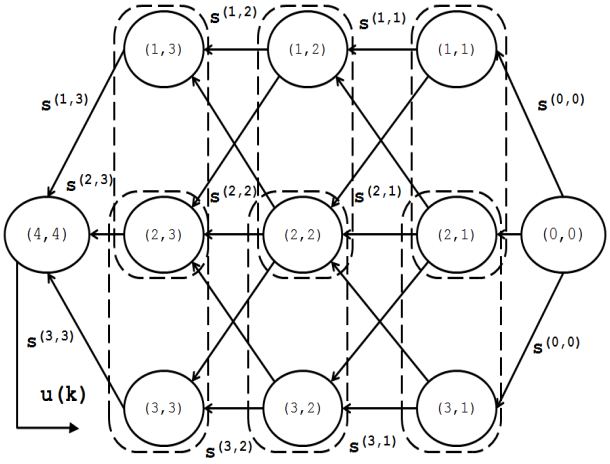

Fig. 3 Graph with $3 \times 3 + 2$ nodes.

2.5 Example

Consider the network in Fig. 3. Some nodes in the network only have one part of the plant model and the other nodes will have the other part. Therefore, nodes $(1,1)$, $(3,1)$, $(1,2)$, $(3,2)$, $(1,3)$ and $(3,3)$ will have the same information about the plant, that is

$$\mathbf{A}_{(1,1)}=\mathbf{A}_{(3,1)}=\mathbf{A}_{(1,2)}=\mathbf{A}_{(3,2)}=\mathbf{A}_{(1,3)}=\mathbf{A}_{(3,3)},\quad \mathbf{B}_{(1,1)}=\mathbf{B}_{(3,1)}=\mathbf{B}_{(1,2)}=\mathbf{B}_{(3,2)}=\mathbf{B}_{(1,3)}=\mathbf{B}_{(3,3)}$$

and

$$\mathbf{W}_{(1,1)(2,1)}=\mathbf{W}_{(3,1)(2,1)}=\mathbf{W}_{(1,2)(2,2)}=\mathbf{W}_{(3,2)(2,2)}=\mathbf{W}_{(1,3)(2,3)}=\mathbf{W}_{(3,3)(2,3)}.$$

On the other hand, the rest of the nodes will have the other part of the information about the plant:

$$\mathbf{A}_{(2,1)}=\mathbf{A}_{(2,2)}=\mathbf{A}_{(2,3)},\quad \mathbf{B}_{(2,1)}=\mathbf{B}_{(2,2)}=\mathbf{B}_{(2,3)}$$

and

$$\mathbf{W}_{(2,1)(1,1)}=\mathbf{W}_{(2,1)(3,1)}=\mathbf{W}_{(2,2)(1,2)}=\mathbf{W}_{(2,2)(3,2)}=\mathbf{W}_{(2,3)(1,3)}=\mathbf{W}_{(2,3)(3,3)}.$$

Moreover, the cooperating couples are: $(1,1)\longleftrightarrow(2,1)$, $(1,2)\longleftrightarrow(2,2)$, $(1,3)\longleftrightarrow(2,3)$, $(3,1)\longleftrightarrow(2,1)$, $(3,2)\longleftrightarrow(2,2)$, and $(3,3)\longleftrightarrow(2,3)$.

So, it is easy to see that:

$$\mathbf{A}_{(1,1)(1,1)}=\mathbf{A}_{(1,2)(1,2)}=\mathbf{A}_{(1,3)(1,3)}=\mathbf{A}_{(3,1)(3,1)}=\mathbf{A}_{(3,2)(3,2)}=\mathbf{A}_{(3,3)(3,3)},$$

$$\mathbf{B}_{(1,1)(1,1)}=\mathbf{B}_{(1,2)(1,2)}=\mathbf{B}_{(1,3)(1,3)}=\mathbf{B}_{(3,1)(3,1)}=\mathbf{B}_{(3,2)(3,2)}=\mathbf{B}_{(3,3)(3,3)},$$

$$\mathbf{A}_{(2,1)(2,1)}=\mathbf{A}_{(2,2)(2,2)}=\mathbf{A}_{(2,3)(2,3)},\quad \mathbf{B}_{(2,1)(2,1)}=\mathbf{B}_{(2,2)(2,2)}=\mathbf{B}_{(2,3)(2,3)},$$

$$\mathbf{A}_{(1,1)(2,1)}=\mathbf{A}_{(3,1)(2,1)}=\mathbf{A}_{(1,2)(2,2)}=\mathbf{A}_{(3,2)(2,2)}=\mathbf{A}_{(1,3)(2,3)}=\mathbf{A}_{(3,3)(2,3)},$$

$$\mathbf{B}_{(1,1)(2,1)}=\mathbf{B}_{(3,1)(2,1)}=\mathbf{B}_{(1,2)(2,2)}=\mathbf{B}_{(3,2)(2,2)}=\mathbf{B}_{(1,3)(2,3)}=\mathbf{B}_{(3,3)(2,3)},$$

$$\mathbf{A}_{(2,1)(1,1)}=\mathbf{A}_{(2,1)(3,1)}=\mathbf{A}_{(2,2)(1,2)}=\mathbf{A}_{(2,2)(3,2)}=\mathbf{A}_{(2,3)(1,3)}=\mathbf{A}_{(2,3)(3,3)}$$

and

$$\mathbf{B}_{(2,1)(1,1)}=\mathbf{B}_{(2,1)(3,1)}=\mathbf{B}_{(2,2)(1,2)}=\mathbf{B}_{(2,2)(3,2)}=\mathbf{B}_{(2,3)(1,3)}=\mathbf{B}_{(2,3)(3,3)}.$$

For example, if

$$\mathbf{A}_{(1,1)}=\begin{bmatrix}\begin{bmatrix}1&1\\0&1\end{bmatrix} & \begin{bmatrix}0&0\\0&0\end{bmatrix}\\ \begin{bmatrix}0&0\\0&0\end{bmatrix} & \begin{bmatrix}0.1&0\\0&0.2\end{bmatrix}\end{bmatrix},\quad \mathbf{A}_{(2,1)}=\begin{bmatrix}\begin{bmatrix}1&1\\0&1\end{bmatrix} & \begin{bmatrix}0&0\\0&0\end{bmatrix}\\ \begin{bmatrix}0&0\\0&0\end{bmatrix} & \begin{bmatrix}0&0\\0&0.1\end{bmatrix}\end{bmatrix},$$

$$\mathbf{B}_{(1,1)}=\begin{bmatrix}\begin{bmatrix}0\\1\\0\\0\end{bmatrix}\end{bmatrix},\quad \mathbf{B}_{(2,1)}=\begin{bmatrix}\begin{bmatrix}0\\1\\0\\0\end{bmatrix}\end{bmatrix},\quad \mathbf{W}_{(1,1)(2,1)}=\begin{bmatrix}\begin{bmatrix}0\\0\end{bmatrix}\\ \begin{bmatrix}0\\0.4\end{bmatrix}\end{bmatrix},\quad \mathbf{W}_{(2,1)(1,1)}=\begin{bmatrix}\begin{bmatrix}0\\0\\0\\1\end{bmatrix}\end{bmatrix}.$$

then, the decentralized models are

$$\mathbf{A}_{(1,1)(1,1)}=\begin{bmatrix}1&1\\0&1\end{bmatrix},\quad \mathbf{A}_{(2,1)(2,1)}=\begin{bmatrix}1&1\\0&1\end{bmatrix},\quad \mathbf{B}_{(1,1)(1,1)}=\begin{bmatrix}0\\1\end{bmatrix},\quad \mathbf{B}_{(2,1)(2,1)}=\begin{bmatrix}0\\1\end{bmatrix},$$

whereas the interacting models are given by:

$$\mathbf{A}_{(1,1)(2,1)} = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.2 \end{bmatrix}, \quad \mathbf{A}_{(2,1)(1,1)} = \begin{bmatrix} 0 & 0 \\ 0 & 0.1 \end{bmatrix}, \quad \mathbf{B}_{(1,1)(2,1)} = \begin{bmatrix} 0 \\ 0.4 \end{bmatrix}, \quad \mathbf{B}_{(2,1)(1,1)} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

3 Description of the Approach

In this section we will describe the control calculation, including the algorithms that gives the nodes to cooperate in order to calculate the control action.

3.1 Control implementation

To implement the control law (3) over the network using packets of the form (4), we will use communication driven cooperative MPC.

In this work we are assuming that pairs of neighbor nodes in the same column can exchange information. In Algorithm 1 we show the cooperation between a pair of nodes. Then, with the CM in (2) and the Algorithm 1, it is possible to calculate a *Feasible Cooperation-Based MPC (FC-MPC)*, as explained in [2].

The calculation of the suboptimal control input, $\mathbf{u}^{*p}_{(i,j)}$, for each iteration p, is performed by solving the FC-MPC problem. So, we will choose the objective function as a linear combination of the individual nodes' objectives, i.e.,

$$J_{(i,j)} = J_{(i^*,j)} = \varpi_{(i,j)} V_{(ij)} + \varpi_{(i^*,j)} V_{(i^*,j)}, \quad \varpi_{(i,j)}, \varpi_{(i^*,j)} > 0, \varpi_{(i,j)} + \varpi_{(i^*,j)} = 1, i^* \in \{i-1, i+1\}$$

The local objective for each cooperative node depends on the value of $\gamma^{(i-1,j-1)}_{(i,j)}(k)$, $\gamma^{(i,j-1)}_{(i,j)}(k)$ and $\gamma^{(i+1,j-1)}_{(i,j)}(k)$. If at least one of them is equal to one, then we will have the following cost function:

$$J_{(i,j)} = V_{(i,j)}(\mathbf{x}'^{p}_{(i,j)}(k), \mathbf{u}'^{p}_{(i,j)}(k); \mathbf{x}_{(i,j)}(k)) = \sum_{t=k}^{\infty} \left\| \mathbf{x}^{p}_{(i,j)}(t|k) \right\|^2_{\mathbf{Q}} + \left\| \mathbf{u}^{p}_{(i,j)}(t|k) \right\|^2_{\mathbf{R}},$$

where $\mathbf{x}'^{p}_{(i,j)}(k) = [\mathbf{x}^{p}_{(i,j)}(k+1|k)^T, \mathbf{x}^{p}_{(i,j)}(k+2|k)^T, ...]^T$, $\mathbf{u}'^{p}_{(i,j)}(k) = [\mathbf{u}^{p}_{(i,j)}(k|k)^T, \mathbf{u}^{p}_{(i,j)}(k+1|k)^T, ...]^T$, and $\mathbf{Q} > 0$ and $\mathbf{R} > 0$ are weighting matrices. To calculate these predictions the CM for the node (i, j) has been used, see (2).

The notation p indicates the iteration number. During each MPC optimization, the state and input trajectories $(\mathbf{x}'_{(i^*,j)}(k), \mathbf{u}'_{(i^*,j)}(k))$ of the interacting node MPC are not updated, so they remain at $(\mathbf{x}'^{p-1}_{(i^*,j)}(k), \mathbf{u}'^{p-1}_{(i^*,j)}(k))$.

On the other hand, if $\gamma^{(i-1,j-1)}_{(i,j)}(k) = \gamma^{(i,j-1)}_{(i,j)}(k) = \gamma^{(i+1,j-1)}_{(i,j)}(k) = 0$, no information about the state has arrived at node (i, j), so an estimation is used instead. The

cost function will be an expected value ([17]):

$$J_{(i,j)} = \mathbf{E}\{\sum_{t=k}^{\infty} \mathbf{x}^p_{(i,j)}(t|k)^T \mathbf{Q}\mathbf{x}^p_{(i,j)}(t|k) + \mathbf{u}^p_{(i,j)}(t|k)^T \mathbf{R}\mathbf{u}^p_{(i,j)}(t|k)\}$$
$$= \sum_{t=k}^{\infty} \hat{\mathbf{x}}^p_{(i,j)}(t|k)^T \mathbf{Q}\hat{\mathbf{x}}^p_{(i,j)}(t|k) + \mathbf{tr}\big(\mathbf{Q}\mathbf{P}_{(i,j)}(k)\big) + \mathbf{u}^p_{(i,j)}(t|k)^T \mathbf{R}\mathbf{u}^p_{(i,j)}(t|k),$$

where $\mathbf{P}_{(i,j)}(k)$ approximates the covariance of $\mathbf{x}_{(i,j)}(k)$ and is calculated as follows:

$$\mathbf{P}_{(i,j)}(k+1) = \mathbf{A}_{(i,j)}\mathbf{P}_{(i,j)}(k)\mathbf{A}^T_{(i,j)} - \Gamma^{(i,j)}(k)\mathbf{P}_{(i,j)}(k)\Gamma^{(i,j)}(k) + \mathbf{D},$$

where

$$\Gamma^{(i,j)}(k) \triangleq \prod_{\substack{\alpha \in \{0,1,\ldots,i-1\} \\ \beta \in \{0,1,\ldots,i-1\}}} \gamma^{(ig^*,jg^*)}_{(\alpha,\beta)}(k)$$

is equal to 1 if and only if $\mathbf{x}_{(i,j)}(k)$ is available at node (i,j) at time $kT + (j-1)\tau$. In the above expression, (ig^*, jg^*) is one of the preceding nodes of (i,j) and is the one that provides the best (the smallest or unique) value of $J_{(i,j)}(k)$.

Remark. The objective function $J_{(i,j)}$ is an approximation of:

$$J_{(i,j)} \approx \mathbf{E}\{V_{(i,j)}(\hat{\mathbf{x}}^{'p}_{(i,j)}(k), \mathbf{u}^{'p}_{(i,j)}(k); \hat{\mathbf{x}}_{(i,j)}(k))\},$$

since $\mathbf{P}_{(i,j)}(k)$ is not the covariance of $\mathbf{x}_{(i,j)}(k)$, but just an approximate value. Therefore the term $\mathbf{tr}\big(\mathbf{Q}\mathbf{P}_{(i,j)}(k)\big)$ is not exact.

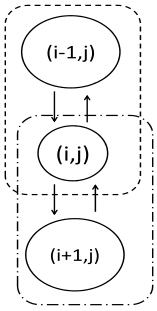

Fig. 4 Cooperative nodes for node (i,j).

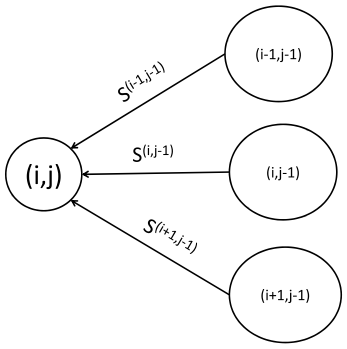

Fig. 5 Packets received by node (i,j).

In the Algorithm 1, the state sequence generated by the input sequence $\mathbf{u}_{(i,j)}$ and initial state $\mathbf{x}_{(i,j)}$ has been represented by $\mathbf{x}^{(\mathbf{u}_{(i,j)};\mathbf{x}_{(i,j)})}_{(i,j)}$. Also, the notation $\hat{\mathbf{x}}$ is representing $[\hat{\mathbf{x}}^T_{(i,j)}, \hat{\mathbf{x}}^T_{(i^*,j)}]^T$.

Algorithm 1 Cooperative MPC algorithm

Given $(\bar{\mathbf{u}}^0_{(\iota,j)}, \mathbf{x}_{(\iota,j)}(k))$, $\mathbf{Q} > 0$, $\mathbf{R} > 0$, $\iota \in \{i, i^*\}$, $i^* \in \{i-1, i+1\}$, $p_{max}(k) \geq 0$, $\varepsilon > 0$, $p \leftarrow 1$, $e_{(\iota,j)} \leftarrow \Phi$ and $\Phi >> 1$.
while $e_{(\iota,j)} > \varepsilon$ for some $\iota \in \{i, i^*\}$ and $p \leq p_{max}(k)$ **do** $\forall \iota \in \{i, i^*\}$
  $\mathbf{u}^{*p}_{(\iota,j)} \in \arg\min_{(\iota,j)} \quad FC-MPC_{(\iota,j)}$
  $\mathbf{u}^{p}_{(\iota,j)} = \varpi_{(\iota,j)} \mathbf{u}^{*p}_{(\iota,j)} + (1-\varpi_{(\iota,j)}) \mathbf{u}^{p-1}_{(\iota,j)}$
  $e_{(\iota,j)} = \left\| \mathbf{u}^{p}_{(\iota,j)} - \mathbf{u}^{p-1}_{(\iota,j)} \right\|$
  for each $\iota \in \{i, i^*\}$ **do**
    The node (ι, j) transmits $\mathbf{u}^{p}_{(\iota,j)}$ to its neighbor
  end for
  if $\gamma^{(i-1,j-1)}_{(i,j)}(k) = 1 \vee \gamma^{(i,j-1)}_{(i,j)}(k) = 1 \vee \gamma^{(i+1,j-1)}_{(i,j)}(k) = 1$ **then**
    $\mathbf{x}^{p}_{(\iota,j)} \leftarrow \mathbf{x}^{(\bar{\mathbf{u}}^{p}_{(i,j)}, \bar{\mathbf{u}}^{p}_{(i^*,j)}; \mathbf{x}_{(\iota,j)})}_{(\iota,j)}$, $\forall \iota \in \{i, i^*\}$
  else
    $\hat{\mathbf{x}}^{p}_{(\iota,j)} \leftarrow \hat{\mathbf{x}}^{(\bar{\mathbf{u}}^{p}_{(i,j)}, \bar{\mathbf{u}}^{p}_{(i^*,j)}; \hat{\mathbf{x}}_{(\iota,j)})}_{(\iota,j)}$, $\forall \iota \in \{i, i^*\}$
  end if
  $p \leftarrow p+1$
end while

Due to the communication constraints, the maximum number of iterations p_{max} is limited. It is also possible to loose information during the cooperation. For these reasons only a suboptimal control input $\mathbf{u}^{*p}_{(i,j)}$ will be available.

Notice that $i^* \in \{i-1, i+1\}$, that means that the node (i, j) can communicate with the nodes $(i-1, j)$ and $(i+1, j)$, see Fig 4. Therefore, node (i, j) will solve two cooperative MPC problems and will have two control values. The control value that the node (i, j) transmits will be the one that provides the lowest cost.

3.2 State estimation

While only the node $c(k)$ will provide the plant input at instant k, in the present formulation all nodes compute local state estimates, $\hat{\mathbf{x}}_{(i,j)}(k)$, by using the data received from one of the preceding nodes, (ig^*, jg^*). This serves as safeguard for instances when the loop is broken due to dropouts.

Since the nodes don't have full information about the plant, they are only able to calculate a part of the state. That means that $\hat{\mathbf{x}}_{(i,j)}(k)$ is not an estimate of the global state of the plant.

In the sequel, we will focus on situations where acknowledgments of plant inputs are "quite reliable". Thus, the state estimates are simply calculated as

$$\begin{aligned} \hat{\mathbf{x}}_{(i,j)}(k) = \mathbf{A}_{(i,j)} \hat{\mathbf{x}}_{(i,j)}(k-1) + \mathbf{B}_{(i,j)} \mathbf{u}_{(i,j)}(k-1) + \mathbf{W}_{(i,j)(i^*,j)} \mathbf{u}_{(i^*,j)}(k-1) + \\ \mathbf{K}^{(i,j)}(k) \big(\mathbf{x}_{(i,j)}(k) - (\mathbf{A}_{(i,j)} \hat{\mathbf{x}}_{(i,j)}(k-1) + \mathbf{B}_{(i,j)} \mathbf{u}_{(i,j)}(k-1) + \mathbf{W}_{(i,j)(i^*,j)} \mathbf{u}_{(i^*,j)}(k-1)) \big), \end{aligned} \tag{5}$$

where $\mathbf{K}^{(i,j)}(k) = \Gamma^{(i,j)}(k)\mathbf{I}$.

In (5), $\mathbf{u}_{(i,j)}(k-1)$ and $\mathbf{u}_{(i^*,j)}(k-1)$ are local plant input estimates. In particular, if $\delta^{(i,j)}(k-1) = 1$, then $\mathbf{u}_{(i,j)}(k-1) = \mathbf{u}^A(k-1)$ and $\mathbf{u}_{(i^*,j)}(k-1) = \mathbf{u}^A(k-1)$, where $\mathbf{u}^A(k-1)$ is the applied control signal in time instant $(k-1)$ by the actuator. On the other hand, at instances where $\delta^{(i,j)}(k-1) = 0$, node (i,j) uses the tentative plant input value transmitted in the second field of the previous packet $\mathbf{s}^{(i,j)}(k-1)$ (if non-empty), or otherwise sets $\mathbf{u}_{(i,j)}(k-1)$ and $\mathbf{u}_{(i^*,j)}(k-1)$ as per (3), see Algorithm 2.

Intuitively, good control performance will be achieved if the state estimation is accurate. Clearly, nodes which are in columns closer to the sensor will have access to more output measurements, see Fig. 1. On the other hand, one can expect that nodes which are physically located in columns closer to the actuator node will on average receive more plant input acknowledgments, thus, have better knowledge of plant inputs.

3.3 Algorithm for dynamic controller placement

Algorithm 2 Dynamic Controller Placement

$k \leftarrow 0$, $\hat{\mathbf{x}}_{(i,j)}(0) \leftarrow 0$, $P_0^{(i,j)} \leftarrow P_0$, $m \leftarrow 0$, $i^* = i-1$ or $i^* = i+1$, the cooperative nodes for (i,j).
while $t \geq 0$ **do** ▷ $t \in \mathbb{R}_{\geq 0}$ is actual time
  while $t \leq kT + m\tau$ **do** ▷ wait-loop
    $m \leftarrow m+1$
  end while
  $\mathbf{P}_{(i,j)}(k+1) \leftarrow \mathbf{A}_{(i,j)}\mathbf{P}_{(i,j)}(k)\mathbf{A}_{(i,j)}^T + \mathbf{D}$
  if $\gamma_{(i,j)}^{(i-1,j-1)}(k) = 0 \wedge \gamma_{(i,j)}^{(i,j-1)}(k) = 0 \wedge \gamma_{(i,j)}^{(i+1,j-1)}(k) = 0$ **then**
    if $\delta^{(i,j)}(k-1) = 1$ **then**
      $\mathbf{u}_{(i,j)}(k), \mathbf{u}_{(i^*,j)}(k), J_{(i,j)}(k) \leftarrow Algorithm \quad 1$
      $\mathbf{s}^{(i,j)}(k) \leftarrow \left([\mathbf{x}_{(i,j)}(k)^T, \mathbf{x}_{(i^*,j)}(k)^T]^T, [\mathbf{u}_{(i,j)}(k)^T, \mathbf{u}_{(i^*,j)}(k)^T]^T, J_{(i,j)}(k)\right)$ ▷ a tentative input
    else
      $\mathbf{s}^{(i,j)}(k) \leftarrow (\emptyset, \emptyset, \emptyset)$
    end if
  end if
  if $\gamma_{(i,j)}^{(i-1,j-1)}(k) = 1 \vee \gamma_{(i,j)}^{(i,j-1)}(k) = 1 \vee \gamma_{(i,j)}^{(i+1,j-1)}(k) = 1$ **then**
    $\mathbf{S} \leftarrow \mathbf{s}^{(i-1,j-1)}(k)$ and/or $\mathbf{s}^{(i,j-1)}(k)$ and/or $\mathbf{s}^{(i+1,j-1)}(k)$ ▷ $\mathbf{S}$ is a set containing all the packets received. If all the packets arrive, $\mathbf{S}$ will contain $\mathbf{s}^{(i-1,j-1)}(k)$, $\mathbf{s}^{(i,j-1)}(k)$ and $\mathbf{s}^{(i+1,j-1)}(k)$
    $(\mathbf{x}^S, \mathbf{u}^S, J^S) \leftarrow \arg\min_l \quad J_l \in \mathbf{S}$
    if $\mathbf{x}^S \neq \emptyset$ **then** ▷ $\mathbf{x}_{(i,j)}(k)$ is available
      $\hat{\mathbf{x}}_{(i,j)}(k) \leftarrow \mathbf{x}_{(i,j)}^S$
      $\mathbf{P}_{(i,j)}(k+1) \leftarrow \mathbf{D}$
    end if
    if $\mathbf{u}^S \neq \emptyset$ **then**
      $\mathbf{u}_{(i,j)}(k) = \mathbf{u}_{(i,j)}^S$
      $\mathbf{u}_{(i^*,j)}(k) = \mathbf{u}_{(i^*,j)}^S$

Algorithm 2 is run at every node (i,j). Since we assume that acknowledgments from the actuator node are, in general, available, but transmissions of packets $\mathbf{s}^{(i,j)}(k)$ are less reliable, nodes in columns closer to the sensor nodes can be expected to have better state estimates than nodes located in columns further down the network. Therefore, preference is given to forward incoming tentative plant input values.

The sensor node $(i,j) = (0,0)$ uses as input: $\mathbf{s}^{(0,0)}(k) = (\mathbf{x}(k), \emptyset, \emptyset), \quad \gamma_{(0,0)}(k) = 1$, where $\emptyset$ means that the field is empty.

This node just passes the information to all the nodes in the first column. The node $(0,0)$, as the node $(M+1, M+1)$, can't calculate control values.

The rest of the nodes in the network can only send information to their three closest neighbors in the following column, except for the lower and uppermost nodes who can only send to two neighbors, see Fig. 2. Therefore, the generic node (i,j) can receive zero, one, two or three (if not border node) packets. In the case that it receives more than one packet (as shown in Fig. 5), it chooses the one with the minimun value of the cost function J.

else
  $\mathbf{u}_{(i,j)}(k), \mathbf{u}_{(i^*,j)}(k), J_{(i,j)}(k) \leftarrow$ *Algorithm* 1
end if
if $\mathbf{u}^S = \emptyset \wedge \delta^{(i,j)}(k-1) = 1$ **then**
  $\mathbf{s}^{(i,j)}(k) \leftarrow \left(\mathbf{x}^S, [\mathbf{u}_{(i,j)}(k)^T, \mathbf{u}_{(i^*,j)}(k)^T]^T, J_{(i,j)}(k)\right)$ ▷ a tentative input
else
  $\mathbf{s}^{(i,j)}(k) \leftarrow (\mathbf{x}^S, \mathbf{u}^S, J^S)$
end if
end if
while $t < kT + (j+1)\tau$ **do** ▷ wait-loop
  $m \leftarrow m+1$
end while
transmit $\mathbf{s}^{(i,j)}(k)$
while $t \leq kT + (M+3)\tau$ **do** ▷ wait-loop
  $m \leftarrow m+1$
end while
if $\delta^{(i,j)}(k) = 1$ **then**
  $\hat{\mathbf{x}}_{(i,j)}(k+1) \leftarrow \mathbf{A}_{(i,j)}\hat{\mathbf{x}}_{(i,j)}(k) + \mathbf{B}_{(i,j)}\mathbf{u}^A_{(i,j)}(k) + \mathbf{W}_{(i,j)(i^*,j)}\mathbf{u}^A_{(i^*,j)}(k)$
else
  $\hat{\mathbf{x}}_{(i,j)}(k+1) \leftarrow \mathbf{A}_{(i,j)}\hat{\mathbf{x}}_{(i,j)}(k) + \mathbf{B}_{(i,j)}\mathbf{u}_{(i,j)}(k) + \mathbf{W}_{(i,j)(i^*,j)}\mathbf{u}_{(i^*,j)}(k)$
end if
$k \leftarrow k+1$
end while

The first column of nodes, the nodes (i,j) with $j = 1$, calculate control values cooperating between them per pairs, as explained in Section 3.1. Each node of that column transmits:

$$\mathbf{s}^{(i,j)}(k) = \left([\mathbf{x}_{(i,j)}(k)^T, \mathbf{x}_{(i^*,j)}(k)^T]^T, [\mathbf{u}_{(i,j)}(k)^T, \mathbf{u}_{(i^*,j)}(k)^T]^T, J_{(i,j)}(k)\right)$$

to its three closest neighbors in the next column of nodes. Subsequent nodes then relay the arrived packets to the actuator node, choosing the ones with minimum $J_{(i,j)}(k)$.

A new tentative control value has to be calculated only in the following cases:

- No packet has arrived from the previous column,

$$\gamma^{(i-1,j-1)}_{(i,j)}(k) = \gamma^{(i,j-1)}_{(i,j)}(k) = \gamma^{(i+1,j-1)}_{(i,j)}(k) = 0,$$

  but the acknowledgment from the actuator has arrived, $\delta^{(i,j)}(k-1) = 1$.
- At least one packet has arrived but all of them have the following structure:

$$\mathbf{s} = (\mathbf{x}, \emptyset, \emptyset),$$

  that means that the state is available but there is no information about the control.

Then $\mathbf{u}_{(i,j)}(k)$ is calculated and the following packet is transmitted to the next column: $\mathbf{s}^{(i,j)}(k) = \left([\hat{\mathbf{x}}_{(i,j)}(k)^T, \hat{\mathbf{x}}_{(i^*,j)}(k)^T]^T, [\mathbf{u}_{(i,j)}(k)^T, \mathbf{u}_{(i^*,j)}(k)^T]^T, J_{(i,j)}(k)\right)$. The estimated state $\hat{\mathbf{x}}_{(i,j)}(k)$ is sent to take into account the cases in which the nodes that receive the packet (they don't calculate control value because they have a packet with $\mathbf{u}_{(i,j)}(k) \neq 0$) have some neighbor that is required to calculated a control value, but it can't estimate the whole state by itself.

4 Simulation example

We consider a system with decentralized models

The decentralized models in this example are

$$\mathbf{A}_{(1,1)(1,1)} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{A}_{(2,1)(2,1)} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{B}_{(1,1)(1,1)} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad \mathbf{B}_{(2,1)(2,1)} = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

and interacting models given by:

$$\mathbf{A}_{(1,1)(2,1)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad \mathbf{A}_{(2,1)(1,1)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad \mathbf{B}_{(1,1)(2,1)} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad \mathbf{B}_{(2,1)(1,1)} = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

where we are considering no noise and $\bar{x_0} = 1$.

The network is as depicted in Fig. 3, with i.i.d. transmission processes and success probabilities $\mathbf{Prob}\{\gamma_{(i,j))}^{(i,j-1)}(k) = 1\} = 0.4$ and $\mathbf{Prob}\{\delta^{(i,j)}(1) = 1\} = 1$.

Fig. 6 shows the empirical distribution of the controller node $c(k)$ obtained by running the algorithm for 100 steps. It is possible to see how 43% of the times, the controller node is located in the last column of nodes, the one closest to the actuator.

Fig. 6 Controller location percentage.

Fig. 7 and 8 compare the plant state trajectory when the algorithm proposed is used with the case in which we locate the controller at the actuator node. The results suggest that the proposed algorithm yields a stable system, while when the controller is at the actuator, the system becomes unstable.

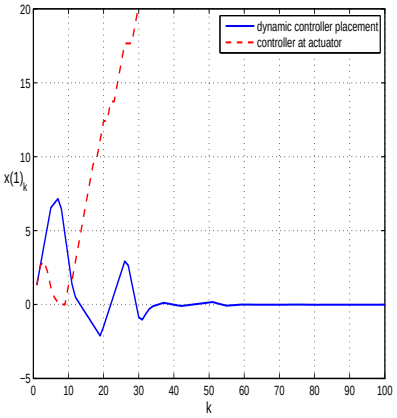

Fig. 7 $x(1)$ trajectory.

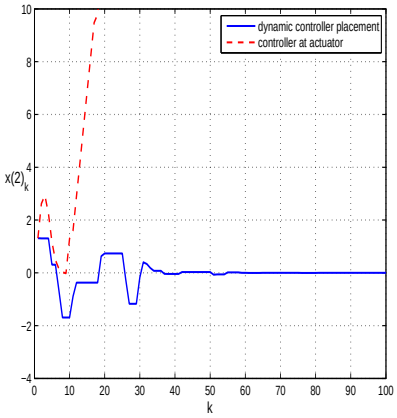

Fig. 8 $x(2)$ trajectory.

5 Conclusions

We have presented a cooperative MPC formulation for NCSs subject to data dropouts. We provide an algorithm that decides which nodes are in charge of the calculation of the the control input, and which ones just relay the received information. This decision depends on transmission outcomes. Once the controller node has been chosen, it interacts with its neighbors over unreliable links solving a cooperative MPC.

Future works may include stability analysis of the proposed architecture. Furthermore, it would be of interest to study a practical application of the proposed method for systems controlled over unreliable networks with time-varying reliability, for example, if there are moving obstacles blocking the nodes, [5].

References

1. A. N. Venkat. *Distributed Model Predictive Control: Theory and Applications*. PhD thesis, University of Wisconsin, Madison, Wisconsin, October 2006.
2. A. N. Venkat, J. B. Rawlings and S. J. Wright. Distributed model predictive control of large-scale systems. In *In Assessment and Future Directions of Nonlinear Model Predictive Control*, pages 591–605. Springer-Verlag Berlin Heidelberg, 2007.
3. A. S. Matveev, and A. V. Savkin. *Estimation and Control over Communication Networks*. *Birkäusser* Boston, 2009.
4. C. L. Robinson and P. R. Kumar. Optimizing controller location in networked control systems with packet drops. *IEEE Journal on Selected Areas in Communications*, 26(4):661–671, May 2008.
5. D. E. Quevedo, A. Ahlén and K. H. Johansson. State estimation over sensor networks with correlated wireless fading channels. *IEEE Transactions on Automatic Control*, Accepted for

publication.
6. D. E. Quevedo, E. I. Silva, and G. C. Goodwin. Subband coding for networked control systems. *International Journal of Robust and Nonlinear Control*, 19(16):1817–1836, November 2009.
7. D. E. Quevedo, K. H. Johansson, A. Ahlén and I. Jurado. Dynamic controller allocation for control over erasure channels. In *Proc. 3rd IFAC Workshop on Distributed Estimation and Control in Networked Systems*, Santa Barbara, CA, 2012.
8. D. Quevedo, J. Østergaard and D. Nešić. Packetized predictive control of stochastic systems over bit-rate limited channels with packet loss. *IEEE Transactions on Automatic Control*, 56(12):2854–2868, December 2011.
9. E. A. Jorswieck, E. G. Larsson, M. Luise and H. V. Poor. Game theory and the flat-fading Gaussian interference channel. *IEEE Signal Processing Magazine (Special Issue on Game Theory in Signal Processing and Communications)*, 26(5):18–27, September 2009.
10. E. Altman, T. Boulogne, R. El-Azouzi, T. Jiménez and L. Wynter. A survey on networking games in telecommunications. *Computer and Operations Research*, 33(2):286–311, February 2006.
11. E. Garone, B. Sinopoli and A. Casavola. LQG control over lossy TCP-like networks with probabilistic packet acknowledgements. *International Journal of Systems Control and Communications*, 2(1,2,3):55–81, January 2010.
12. F. Xiao and L. Wang. Consensus protocols for discrete-time multi-agent systems with time-varying delays. *Automatica*, 44(10):2577–2582, September 2008.
13. G. N. Nair, F. Fagnani, S. Zampieri, and R. J. Evans. Feedback control under data rate constraints: An overview. *Proceedings of the IEEE (Special Issue on Technology of Networked Control Systems)*, 95(1):108–137, January 2007.
14. J. B. Rawlings and B. T. Stewart. Coordinating multiple optimization-based controllers: New opportunities and challenges. *Journal of Process Control*, 18(9):839–845, October 2008.
15. J. B. Rawlings and D. Q. Mayne. *Model Predictive Control: Theory and Design*. Nob Hill Publishing, LLC, 2009.
16. J. Chen, K. H. Johansson, S. Olariu, I. C. Paschalidis and I. Stojmenovic. Guest editorial special issue on wireless sensor and actuator networks. *IEEE Transactions on Automatic Control*, 56(10):2244–2246, October 2011.
17. K. J. Åström. *Introduction to Stochastic Control Theory*. Dover Publications, Inc., 2006.
18. O. C. Imer and S. Yüksel and T. Başar. Optimal control of LTI systems over unreliable communication links. *Automatica*, 42(9):1429–1439, September 2006.