2009 American Control Conference
Hyatt Regency Riverfront, St. Louis, MO, USA
June 10-12, 2009

A Distributed MPC scheme with Low Communication Requirements

J. M. Maestre, D. Muñoz de la Peña and E. F. Camacho

Abstract— **In this work, we consider the problem of controlling two linear systems coupled through the inputs. We propose a novel distributed model predictive control method based on game theory in which two different agents communicate in order to find a cooperative solution to the centralized control problem. We assume that each agent only has partial information of the model and the state of the system. The class of systems considered arises naturally in multi-input multi-output processes in which a transfer function model is obtained using standard identification techniques. The performance and the robustness of the proposed control scheme with respect to data losses in the communications are illustrated by extensive simulations.**

I. INTRODUCTION

Model predictive control (MPC) is a control strategy that has been successfully applied to process control, specially to constrained multivariable systems exhibiting dead times [1]. In general, MPC cannot be applied to large scale systems due to the computational requirements or to the impossibility of obtaining a centralized model of the whole system. Typical examples of large scale systems are transportation systems such us traffic, water or power networks [2]. In addition, there is an increasing interest in networked control systems (NCS) made up of different interconnected processes and controllers [9], [10]. One way to address this class of control problems in the MPC framework is to develop distributed model predictive control (DMPC) schemes, in which the system is controlled by agents with limited capabilities which implement a control law based on a reduced model of the system and on partial state information. In general the computational burden of these distributed schemes is lower than the one corresponding to the centralized MPC, but the performance of the closed-loop system depends on the decisions that all the agents take, so cooperation and communication policies become very important issues.

In the context of distributed MPC design, several distributed MPC schemes have been proposed in the literature that deal with the coordination of separate MPC controllers that communicate in order to obtain optimal input trajectories in a distributed manner; see [6], [13] for reviews of results in this area. In [2] an algorithm based on Lagrangian relaxation is presented. In [3], sufficient conditions that guarantee stability of a class of distributed controllers are given. In [4] a DMPC algorithm was applied to a quadruple tank system. In [5], basic collaboration algorithms are provided with an extensive list of conditions to ensure convergence and stability. In [14], the problem of distributed control of dynamically coupled nonlinear systems that are subject to decoupled constraints was considered. In [15], [16], the effect of the coupling was modeled as a bounded disturbance compensated using a robust MPC formulation. In [17] distributed MPC of decoupled systems (a class of systems of relevance in the framework of multi-agents systems) was studied. In [12] Lyapunov based distributed and decentralized schemes for nonlinear systems were presented. Finally in [7] a DMPC framework is proposed with guaranteed stability and optimality properties.

However, for low-resource systems many of the distributed control schemes that can be found in the literature are not suitable due to limitations in the communication capabilities. In wireless sensor and actuator networks, communication is a very valuable and scarce resource and its desirable to have algorithms that use few communication steps. In this paper we propose a DMPC algorithm for two agents based on game theory with reduced communication requirements, specially suited for low-resource systems. Game theory is a theoretical framework that allows one to study the problem of cooperation of different agents with, maybe, conflicting control goals, from a mathematical point of view. In the proposed scheme, the coordination problem between the agents is reduced to a game in which they have to choose one out of three options, and only two communication cycles are needed to reach an agreement. The proposed algorithm provides a feasible solution to the equivalent centralized control problem. In this context it is very important to have control algorithms capable to cope with communications errors; the stability of the overall system may depend on it [8], [11]. In a real distributed environment, errors in the communications and delays in the packets transmissions should be expected. Motivated by this issues, the algorithm's robustness against communication failure, a critical issue in low-resource distributed applications, is also studied though extensive simulation.

II. PROBLEM FORMULATION

In this work we consider the following class of distributed linear systems that consists of two different subsystems coupled with the neighbor subsystem through the inputs:

$$\begin{aligned} x_1(k+1) &= A_1x_1(k) + B_{12}u_1(k) + B_{21}u_2(k) \\ x_2(k+1) &= A_2x_2(k) + B_{21}u_1(k) + B_{22}u_2(k) \end{aligned} \tag{1}$$

where $x_i \in \mathbb{R}^{n_i},\ i = 1,2$ are the states of each subsystem and $u_j \in \mathbb{R}^{m_j},\ i = 1,2$ are the different inputs. This class

Financial support from the European Commission, contract number INFSO-ICT-223854, is gratefully acknowledged.

The authors are with the Departmento de Ingeniería de Sistemas y Automática Universidad de Sevilla, Sevilla 41092, Spain, {pepemaestre,davidmps,eduardo}@cartuja.us.es.

of systems are of relevance when identifications techniques are used to obtain the transfer function of a MIMO process.

The objective of a centralized MPC controller is to minimize a cost function that depends on the predicted trajectories of both states and inputs. At each sampling time, the centralized MPC controller optimizes over the set of input trajectories of length N, the prediction horizon. The optimization variables are

$$\begin{array}{l} U_1 = [u_1(0)^T \; u_1(2)^T \dots u_1(N-1)^T]^T \\ U_2 = [u_2(0)^T \; u_2(2)^T \dots u_2(N-1)^T]^T. \end{array}$$

The MPC optimization problem is defined as follows

$$\begin{array}{ll} \min\limits_{U_1,U_2} & J(U_1,U_2,x_1,x_2) \\ & x_1(k+1) = A_1x_1(k) + B_{12}u_1(k) + B_{21}u_2(k) \\ & x_2(k+1) = A_2x_2(k) + B_{21}u_1(k) + B_{22}u_2(k) \\ & x_1(0) = x_1, \; x_2(0) = x_2 \end{array} \tag{2}$$

where x_1 and x_2 are the current state of the whole system and

$$\begin{array}{rl} J = & \sum\limits_{k=0}^{N-1} x_1(k)^TQ_1x_1(k) + x_2(k)^TQ_2x_2(k) \\ & +u_1(k)^TR_1u_1(k) + u_2(k)^TR_2u_2(k). \end{array}$$

The centralized MPC controller is based on a receding horizon scheme; that is, at each sampling time, the current state of the system x_1, x_2 is received from the sensors, problem (2) is solved and the optimal input trajectory of length N is obtained. The first inputs are applied, and the procedure is repeated the next sampling time. In general, for large scale problems, it might be computationally expensive to solve (2). In the next section we proposed a distributed control scheme in which the inputs are decided solving a sequence of reduced optimization problems based on the model of each subsystem.

III. Distributed MPC

An important issue for distributed control schemes is the computational and communicational burden that the agents have to bear in order to control the system. Centralized MPC solves a single large-scale problem and has no need of communications. The goal of distributed and decentralized systems is to obtain the same closed-loop performance as the centralized MPC solving a sequence of lower complexity optimization problems. A control system is decentralized if there is not communication among the agents. This is the worst scenario from the performance point of view because each agent has to cope alone with its control problem with the risk that the absence of coordination in the agents' decisions may lead to the instability of the system. The control system is distributed if there is communication between agents. The degree of communication depends on the control problem and the communication constraints. In this section we present a distributed MPC controller based on a cooperative game scheme between two different agents. We assume that each agent has access only to partial model information; that is, agent 1 decides u_1 and has information of the dynamics and the measurements of x_1, while agent 2 decides u_2 and has information of the dynamics and the measurements of x_2. This implies that in order to cooperate to minimize a global cost function, both agents have to communicate.

Each agent defines a local cost function based on the predicted trajectories of its state and input; that is,

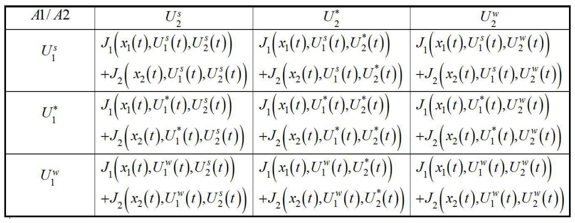
$$\begin{array}{l} J_1 = \sum\limits_{k=0}^{N-1} x_1(k)^TQ_1x_1(k) + u_1(k)^TR_1u_1(k) \\ J_2 = \sum\limits_{k=0}^{N-1} x_2(k)^TQ_2x_2(k) + u_2(k)^TR_2u_2(k). \end{array}$$

The objective of the DMPC scheme is to minimize the global cost function $J = J_1 + J_2$. To this end, each agent solves a sequence of reduced dimension optimization problems based on the model of its subsystem assuming a given fixed input trajectory for its neighbor. In order to describe the algorithm, we use the following notation:

- U_i: Future input trajectory of agent i with $i = 1, 2$.
- U_{nei}: Future input trajectory of the neighbor of agent i; that is, $U_{ne1} = U_2$ and $U_{ne2} = U_1$.
- U_i^s: Shifted optimal input trajectory of agent i; that is, if at a given sampling time the optimal input trajectory of agent i, denoted U_i^{dmpc} is
$$U_1^{dmpc} = [u_1^*(0)^T \; u_1^*(1)^T \dots u_1^*(N-1)^T]^T$$
then
$$U_1^s = [u_1^*(1)^T \; u_1^*(2)^T \dots u_1^*(N-1)^T \; 0]^T.$$
Note that we use the index k to define the time steps of the optimization problems, not the real sampling times.

The proposed DMPC algorithm is the following:

1) Each agent i receives its corresponding partial state measurement x_i.
2) Each agent i minimizes J_i assuming that the neighbor keeps applying the optimal trajectory evaluated at the previous time step; that is, $U_{nei} = U_{nei}^s$.
$$\begin{array}{rl} U_i^* = \min\limits_{U_i} & J_i(U_i, U_{nei}^s, x_i) \\ & x_i(k+1) = A_ix_i(k) + B_{ii}u_i(k) \\ & \qquad\qquad +B_{i,nei}u_{nei}(k) \\ & x_i(0) = x_i. \end{array}$$
Note that in this optimization problem the free variable is U_i (the neighbor input trajectory is fixed).
3) Each agent i minimizes J_i optimizing the neighbor input assuming that he applies the input trajectory computed in the previous optimization problem U_i^*.
$$\begin{array}{rl} U_{nei}^w = \min\limits_{U_{nei}} & J_i(U_i^*, U_{nei}, x_i) \\ & x_i(k+1) = A_ix_i(k) + B_{ii}u_i(k) \\ & \qquad\qquad +B_{i,nei}u_{nei}(k) \\ & x_i(0) = x_i \end{array}$$
Note that in this optimization problem the free variable is U_{nei} (the input trajectory U_i is fixed). Solving this optimization problem, agent i defines an input trajectory for its neighbor that optimizes its local cost function J_i.



A1/A2	U_2^s	U_2^*	U_2^w
U_1^s	$J_1(x_1(t),U_1^s(t),U_2^s(t)) + J_2(x_2(t),U_1^s(t),U_2^s(t))$	$J_1(x_1(t),U_1^s(t),U_2^*(t)) + J_2(x_2(t),U_1^s(t),U_2^*(t))$	$J_1(x_1(t),U_1^s(t),U_2^w(t)) + J_2(x_2(t),U_1^s(t),U_2^w(t))$
U_1^*	$J_1(x_1(t),U_1^*(t),U_2^s(t)) + J_2(x_2(t),U_1^*(t),U_2^s(t))$	$J_1(x_1(t),U_1^*(t),U_2^*(t)) + J_2(x_2(t),U_1^*(t),U_2^*(t))$	$J_1(x_1(t),U_1^*(t),U_2^w(t)) + J_2(x_2(t),U_1^*(t),U_2^w(t))$
U_1^w	$J_1(x_1(t),U_1^w(t),U_2^s(t)) + J_2(x_2(t),U_1^w(t),U_2^s(t))$	$J_1(x_1(t),U_1^w(t),U_2^*(t)) + J_2(x_2(t),U_1^w(t),U_2^*(t))$	$J_1(x_1(t),U_1^w(t),U_2^w(t)) + J_2(x_2(t),U_1^w(t),U_2^w(t))$

Fig. 1. Cost function table used for the decision making.

4) Both agents communicate. Agent 1 sends U_1^* and U_2^w to agent 2 and receives U_2^* and U_1^w.
5) Each agent evaluates the local cost function J_i for each the nine different possible combination of input trajectories; that is $U_1 \in \{U_1^s, U_1^w, U_1^*\}$ and $U_2 \in \{U_2^s, U_2^w, U_2^*\}$
6) Both agents communicate and share the information of the value of local cost function for each possible combination of input trajectories. In this step, both agents receive enough information to take a cooperative decision.
7) Each agent applies the input trajectory that minimizes $J = J_1 + J_2$. Because both agents have access to the same information after the second communication cycle, both agents will chose the same optimal input sets. We denote the chosen set of input trajectories
$$\{U_1^{dmpc}, U_2^{dmpc}\}$$
8) The first input of each optimal sequence is applied and the procedure is repeated the next sampling time.

From a game theory point of view, both agents are playing a cooperative game. This game can be synthesized in strategic form by a 3x3 matrix. Every row represents a possible decision of agent 1 and every column represents the possible decisions for agent 2. The resulting cells contain the sum of the cost functions of both agents. At each time step, the option that yields a lower global cost is chosen. These nine possibilities are shown in the next table.

Remark 1: A qualitative analysis can be done for the proposed solutions at each step for the agent i:

- U_i^s: Stable option. The agent has de possibility to maintain its actuation. This can be a good option because U_j^* is calculated supposing that $U_i = U_i^s$. This also allows the system globally to stay stable once a global J minimum has been reached.
- U_i^*: Selfish option. This option offers an improvement in J_i if the rest of the system's manipulated variables stay unchanged. It can be a good choice to reduce J if the rest of the agents have reached a steady state actuation.
- U_i^w: Altruist option. This option offers the best improvement for the neighbor agent. whose wish is $U_i = U_i^w$ when it takes U_j^* . The agent i sacrifices its own welfare in order to improve the performance of its neighbor.

Remark 2: The proposed scheme can be extended to deal with a N agents, however, in order to build a global cost table to take a cooperative decision, the complexity in general grows exponentially. In order to reduce the complexity, the structure of the system may be exploited taking into account that all the input may not affect all the outputs. Also, in general not all the possible cooperation options are employed with the same frequency, so is possible to reduce further the complexity by not taking into account the less frequent options.

Remark 3: In this work we have not considered state constraints, however, the results can be extended to constrained systems if the optimization problem is modified appropriately.

Remark 4: In general, the minimum number of communication steps needed for a cooperating control scheme is two. In the first step each agent informs of its intentions to its neighbors and during the second it can confirm if it accepts its neighbors' intentions.

Remark 5: Ideally it would be desirable that each agent could access the whole state of the system. However, most of the times is impossible to reach such a communication level due to constraints in the transmission rate, the channel capacity or the size of the system. Moreover, sharing all the information can be unpractical because not all the variables in the system are relevant for all the agents. This is one of the reasons why in general agents only communicate with their neighbors; that is, those agents whose variables are relevant for them.

A. Communication errors

The proposed algorithm assumes flawless communications between both agents. In a real distributed environment, errors in the communications and delays in the packets transmission should be expected. In this section we modify the proposed strategy to take into account data losses and delays in the packet transmissions. To simplify the notation, we assume that an error in the communication link will affect the transmissions in both ways, so there is no possibility that only one of the agents is affected by a fault.

To model data losses and possible delays, we assume that the possibility of flawless communications is given by the parameter $reliability \in [0, 1]$. This parameter characterizes the quality communication network. In the following section, different simulations for different values of this parameter are done.

In the original algorithm the agents chose among three options for the control signal (U_i^s, U_i^*, U_i^w) with the goal of minimizing J. When data losses occur, the agents do not receive U_i^w or the information needed to build the global cost table. In this case, each agent must decide wether to keep applying the last optimal input trajectory U_i^s, or act selfishly and try to minimize its local cost function choosing U_i^*. In order to test the robustness of the proposed approach in the worst possible case, we assume that when communication errors occur, each controller operates in a decentralized way, applying U_i^*.

Remark 6: Note that as the parameter $reliability$ tends to zero, the amount of information shared by the agents de-

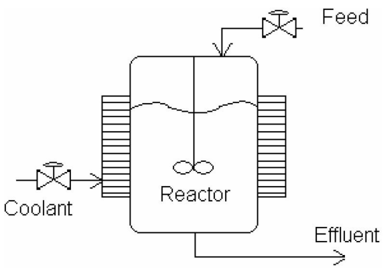


Fig. 2. Continuously stirred tank reactor (CSTR)

creases, and the controller tends to operate in a decentralized manner.

IV. Simulations

To demonstrate the proposed controller, we use the linearized model of a continuously stirred tank reactor (CSTR) presented in [1]. The linearized process around a given equilibrium point is described in continuous time by the following transfer matrix

$$\begin{bmatrix} Y_1(s) \\ Y_2(s) \end{bmatrix} = \begin{bmatrix} \frac{1}{1+0.7s} & \frac{5}{1+0.3s} \\ \frac{1}{1+0.5s} & \frac{2}{1+0.5s} \end{bmatrix} \begin{bmatrix} U_1(s) \\ U_2(s) \end{bmatrix},$$

where the manipulated variables U_1 and U_2 are respectively the flow rate and the flow of coolant in the jacket. The controlled variables Y_1 and Y_2 are respectively the effluent concentration and the reactor temperature, see figure 2. The sampling time is defined as $T_s = 0.03s$.

The control objective is to track a given constant reference from a random initial state. We first design a centralized MPC scheme that decides both inputs simultaneously. The MPC optimization problem that has been used for the simulations is based on minimizing the following cost function using the linearized discrete model of the process

$$\begin{aligned} J = & \sum_{k=0}^{N-1} (ref_1(k) - y_1(k))^T W_{y,1} (ref_1(k) - y_1(k)) \\ & + (ref_2(k) - y_2(k))^T W_{y,2} (ref_2(k) - y_2(k)) \\ & + \Delta u_1(k)^T W_{\Delta u,1} \Delta u_1(k) \\ & + \Delta u_2(k)^T W_{\Delta u,2} \Delta u_2(k), \end{aligned}$$

where $ref_i(k)$ is the reference signal for the controlled variables Y_1 and Y_2. For this simulations we have not considered constraints on the input or the outputs. The following values were used for the controller parameters:

$$\begin{aligned} & N = 5 \\ & ref_1 = 0 \\ & ref_2 = 0 \\ & W_{y,1} = W_{y,2} = 1 \\ & W_{\Delta u,1} = W_{\Delta u,2} = 0.05 \end{aligned} \tag{3}$$

The centralized controller provides the optimal solution from the performance point of view; that is, if the agents could communicate an infinite number of cycles, the solution would converge to the centralized one. We will compare a decentralized and the proposed scheme with this controller.

The following parameters are used to compare the performance of each controller:

- λ: Convergence rate of the global cost function. It is computed as the smaller value such that the following constraint holds
$$J(kT_s) \leq J_0 \cdot \lambda^k, \lambda > 0 \tag{4}$$
where $J(t)$ is the value of the global cost function evaluated at time t for the decided future input trajectories. If the controlled system is unstable then $\lambda > 1$.
- J_v: Number of sampling times required in order to get a relative error below 5%, where the relative error is defined as
$$E_{ri} = \left| \frac{ref_i - y_i}{ref_i} \right| \cdot 100. \tag{5}$$

Over 20 simulations of the system in closed-loop with the centralized controller were done with the references given before and different initial states, half of them with $k_{max} = 100$ and the other half with $k_{max} = 300$. The average performance parameters were obtained for the centralized controller:

$$\begin{aligned} & \lambda = 0.77 \\ & \text{Jv} = 13. \end{aligned}$$

These values will be used to compare the performance of the decentralized and the distributed schemes.

A. Decentralized MPC

We consider that the CSTR is controlled by two different agents. Agent 1 controls the flow rate U_1 based on the measurements of the Y_1, while agent 2 controls U_2 based on the measurements of Y_2. Each agent has an incomplete model of the system; that is, they only know the first row of the system model (how their measured output is affected by each of the inputs). A decentralized MPC scheme is based on the idea that each agent tries to control its own subsystem without communicating with the other agent. Each agent tries to minimize a local cost function. For agent 1 the local cost function is

$$\begin{aligned} J_1 = & \sum_{k=0}^{N-1} (ref_1(k) - y_1(k))^T W_{y,1} (ref_1(k) - y_1(k)) \\ & + \Delta u_1(k)^T W_{\Delta u,1} \Delta u_1(k). \end{aligned}$$

and for agent 2 the local cost function is:

$$\begin{aligned} J_2 = & \sum_{k=0}^{N-1} (ref_2(k) - y_2(k))^T W_{y,2} (ref_2(k) - y_2(k)) \\ & + \Delta u_2(k)^T W_{\Delta u,2} \Delta u_2(k). \end{aligned}$$

At each time step, agent 1 receives Y_1 and finds the optimal sequence of inputs such that J_1 is minimized assuming that $U_2 = 0$, the equilibrium input when the reference is $ref_1 = ref_2 = 0$. Agent 2 follows the same protocol. For this particular system the decentralized controller is not able to stabilize the system. These simulations demonstrate that even for a simple system, when different agents control the same system, if a cooperation scheme is not used, the system may become unstable.

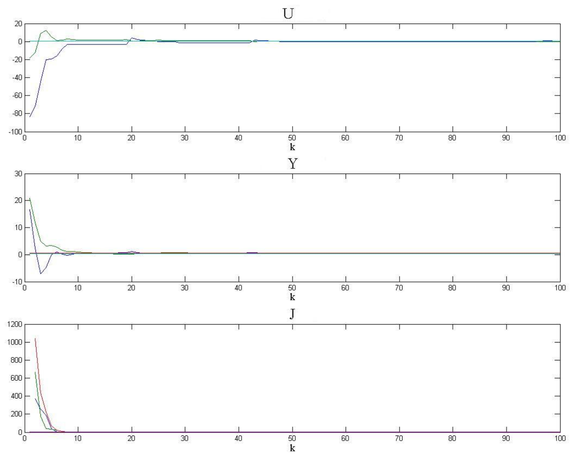


Fig. 3. Trajectories of the system in closed-loop with the proposed DMPC and $reliability = 1$.

B. Distributed MPC

In this section we carry out a set of simulations using the proposed DMPC scheme. Each agent tries to minimize local cost function J_i presented in the previous section as in the decentralized scheme. In this case however, agents do communicate and try to minimize the sum of their optimization functions following the proposed DMPC scheme.

In this case, if there are no data losses or delays ($reliability = 1$), the proposed controller scheme is able to stabilize the closed-loop system. We carried out over 20 simulations with different initial states and constant references. Figure 3 shows one of these simulations. For this set of simulations the performance parameters were $J_v = 41.1458$ and $\lambda = 0.8858$. It can be seen that the performance of the distributed scheme is worst than the one of the centralized controller (although much better than the decentralized scheme which is not able to stabilize the system). As mentioned before, the centralized scheme is the best possible controller from the communication point of view.

C. Communication errors

In order to test the robustness of the proposed DMPC with respect to communications errors, a set of simulations with different $reliability$ values were carried out. for each value $reliability$ over 20 simulations were done. The results obtained are shown tables 2 and 3. Figure 4 shows the dependence of the performance parameters J_v and λ on the network quality parameter $reliability$.

Table 2: λ and J_v for $reliability \in [0.5, 0.9]$

	$reliability$				
	0.9	0.8	0.7	0.6	0.5
$\lambda_{k=100}$	0.8929	0.8976	0.9098	0.9315	0.9902
$J_{v\,k=100}$	44.6375	48.025	64.1375	76.575	84.15
$\lambda_{k=300}$	0.9598	0.9599	0.9625	0.969	0.984
$J_{v\,k=300}$	45.375	52.8125	71.3125	90.9375	129.625

Table 3: λ and J_v for $reliability \in [0.1, 0.4]$

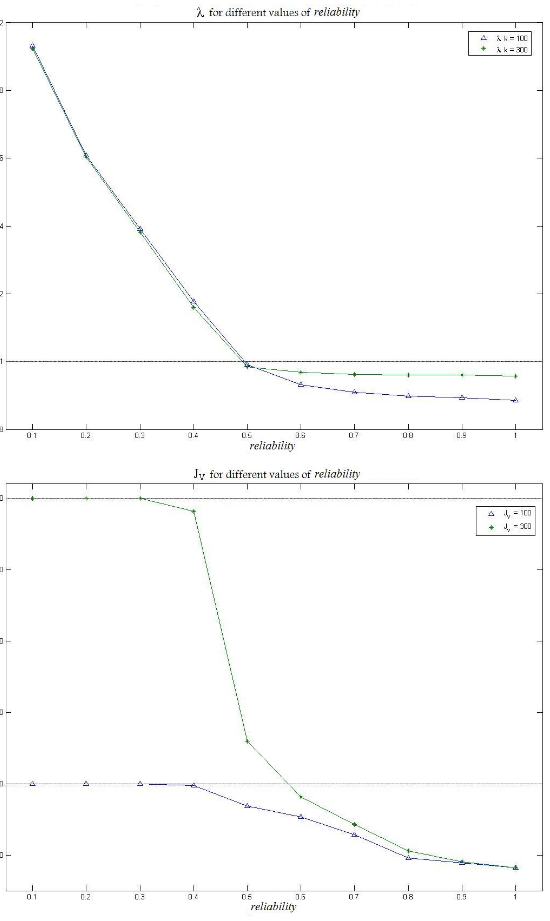


Fig. 4. λ and J_v dependence on the parameter $reliability$.

	$reliability$			
	0.4	0.3	0.2	0.1
$\lambda_{k=100}$	1.1767	1.3917	1.6086	1.9318
$J_{v\,k=100}$	98.8625	100	100	100
$\lambda_{k=300}$	1.1604	1.3818	1.6041	1.9235
$J_{v\,k=300}$	291.0313	300	300	300

Notice that the value of k affects the value of the comparison parameters. An increment in the value of λ is found when k increases if the system stays stable ($\lambda < 1$). This is due to the fact that if the system has reached the desired value for the controlled variables during the first k_1 sampling times, the evolution during the k_2 following time steps wont be significant. As λ is calculated as a function of the total number of the simulation steps the final k_2 time steps will only degrade quantitatively its value, specially if k_2 is in the same order of magnitude than k_1.

On the other hand, it is also observed an increment in J_v with k. The reason for this is that if a simulation fails to regulate the average error below the 5% it will have $J_v = k_{max}$.

As the $reliability$ increases, the performance parameters tend to the ones obtained in the flawless communication simulations. The simulation results also show that depending on the value of $reliability$, the DMPC is able to stabilize the closed-loop system or not. For $reliability \leq 0.5$, the performance parameter is $\lambda < 1$, this implies that the closed-loop system is stable. However, if more than 50% of the

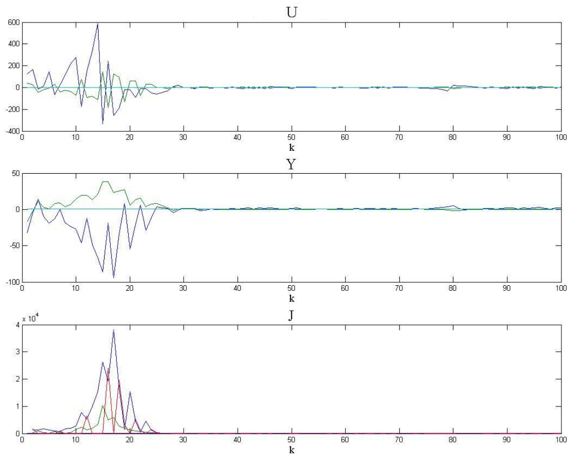


Fig. 5. Trajectories of the system in closed-loop with the proposed DMPC and $reliability$ = 0.5.

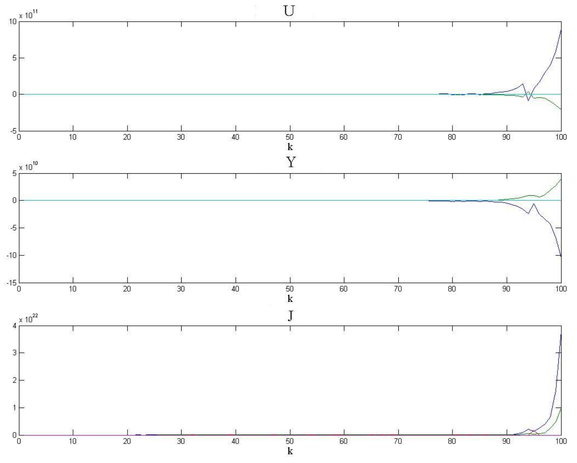


Fig. 6. Evolution of the system with $reliability = 0.2$.

communications fail, then the DMPC is not able to stabilize the closed-loop system. These results demonstrate that for this particular example, when the communication network becomes faulty, the proposed controller tends to operate in a decentralized manner, and hence, is not able to stabilize the system.

V. CONCLUSIONS

In this work we have proposed a novel distributed MPC algorithm based on game theory for a class of systems controlled by two agents. The proposed controller only needs two communication steps in order to obtain a cooperative solution to the centralized optimization problem. Each agent solves an optimization problem that only depends on its local model and partial state information. After sharing information about the local cost, the agents chose the solution that yields the best global performance among a set of suboptimal possibilities. The options are suboptimal because each agent has an incomplete view of the system and they propose the best solutions from their point of view. The algorithm was modified in order to take into account delays and errors in the communications. The proposed algorithm has low communication and computational burdens and provides a feasible solution to the centralized problem. The performance and the robustness of the proposed algorithm with respect to delays and errors in the communications were tested in simulation.

VI. ACKNOWLEDGMENTS

The authors thank F. J. Muros for his help carrying out the CSTR simulations.

REFERENCES

[1] E.F. Camacho and C. Bordons, *Model Predictive Control in the Process Industry*,Berlin, Germany: Springer-Verlag, 1995.
[2] Rudy R. Negenborn, Bart De Schutter, and Hans Hellendoorn., "Multi-Agent Model Predictive Control of Transportation Networks."*Proceedings of the 2006 IEEE International Conference on Networking, Sensing and Control (ICNSC 2006)*,Ft. Lauderdale, Florida, pp. 296–301, Apr. 2006.
[3] Li, S., Zhang, Y., and Zhu., "Nash-optimization enhanced distributed model predictive control applied to the Shell benchmark problem", *Inf. Sci. Inf. Comput. Sci*, 170, 2-4 (Feb. 2005), 329-349.
[4] M. Mercangoz, F. J. Doyle., "Distributed Model Predictive Control of an Experimental four-tank system", *Journal of Process Control*, , Volume 17, Issue 3, pp. 297-308.
[5] E. Camponogara, "Controlling Networks with Collaborative Nets", *PhD Thesis*, Carnegie Mellon University. September 2000.
[6] E. Camponogara et al., "Distributed Model Predictive Control", *IEEE Control Systems Magazine, Feb. 2002, pp. 44-52*, 2002.
[7] Aswin N. Venkat, James B. Rawling and Stephen J. Wright., "Stability and optimality of distributed model predictive control", *Proceedings of the 44th IEEE Conference on Decision and Control.*, 2005.
[8] W. Zhang, M. S. Branicky, and S. M. Phillips, "Stability of networked control systems", *IEEE Control Systems Magazine, vol. 21, pp. 8489* , 2001.
[9] T. C. Yang, Networked control systems: a brief survey,*IEE Procedings-Control Theory and Applications, vol. 153, pp. 403412*, 2006.
[10] P. Neumann, Communication in industrial automation - what is going on? *Control Engineering Practice, vol. 15, pp. 13321347*, 2007.
[11] Muñoz de la Pena, D. and P. D. Christofides, "Lyapunov-Based Model Predictive Control of Nonlinear Systems Subject to Data Losses",*IEEE Trans. Autom. Contr., regular paper in press*.
[12] Liu, J., D. Munoz de la Pena and P. D. Christofides, "Distributed Control System Design Using Lyapunov-Based Model Predictive Control", *Proceedings of International Workshop on Assessment and Future Directions of Nonlinear Model Predictive Control, 12 pages, Pavia, Italy*, 2008.
[13] J. B. Rawlings and B. T. Stewart, Coordinating multiple optimization-based controllers: New opportunities and challenges,/it J. Proc. Contr., in press.
[14] W. B. Dunbar, Distributed receding horizon control of dynamically coupled nonlinear systems,*IEEE Transactions on Automatic Control, vol. 52, pp. 12491263*, 2007.
[15] A. Richards and J. P. How, Robust distributed model predictive control, *International Journal of Control, vol. 80, pp. 15171531*, 2007.
[16] D. Jia and B. Krogh, Min-max feedback model predictive control for distributed control with communication,*in Proceedings of the American Control Conference, Anchorage, 2002, pp. 45074512*, 2002.
[17] T. Keviczky, F. Borrelli, and G. J. Balas, Decentralized receding horizon control for large scale dynamically decoupled systems,*Automatica, vol. 42, pp. 21052115*, 2006.
[18] Y. Sun and N. H. El-Farra, Quasi-decentralized model-based networked control of process systems, *Computers and Chemical Engineering, vol. 32, pp. 20162029*, 2008.