ELSEVIER
Volume 24 • Issue 4 • April 2014
ISSN 0959-1524
Journal of Process Control
IFAC
A Journal of IFAC
The International Federation of Automatic Control
www.elsevier.com/locate/jprocont

Journal of Process Control 24 (2014) 424–434

Contents lists available at ScienceDirect

Journal of Process Control

journal homepage: www.elsevier.com/locate/jprocont

Realization issues, tuning, and testing of a distributed predictive control algorithm

Giulio Betti, Marcello Farina*, Riccardo Scattolini

Dipartimento di Elettronica, Informazione e Bioingengeria, Politecnico di Milano, Piazza Leonardo da Vinci, 32, I-20133 Milan, Italy

ARTICLE INFO

Article history:
Received 12 July 2013
Received in revised form 17 February 2014
Accepted 17 February 2014
Available online 15 April 2014

Keywords:
Model predictive control
Distributed control
Linear systems
Disturbance rejection

ABSTRACT

A non-iterative, non-cooperative distributed state-feedback control algorithm based on neighbor-to-neighbor communication, named distributed predictive control (DPC), has been recently proposed in the literature for constrained linear discrete-time systems, see [15,14,2,4]. The theoretical properties of DPC, such as convergence and stability, its extensions to the output feedback and tracking problems, and applications to simulated plants have been investigated in these papers. However, for a practical use of DPC some realization issues are still open, such as the automatic selection of some tuning parameters, the initialization of the algorithm, or its response to unexpected disturbances which could lead to the lack of the recursive feasibility, a fundamental property for any model predictive control (MPC) technique.

This paper presents novel solutions to all these issues, with the goal to make DPC attractive for industrial and practical applications. Three realistic simulation examples are also discussed to evaluate the proposed numerical algorithms and to compare the performances of DPC to those of a standard centralized MPC algorithm.

1. Introduction

Due to the growing complexity of process plants and to the increasing number of networks of systems, in the last decades researchers have been putting huge efforts in the field of decentralized and distributed control [26,19]. Distributed solutions seem to be very promising with respect to decentralized schemes, because they allow one to take advantage of information transmission between the local controllers, see e.g. [18], and do not require the computational and communication loads of centralized solutions. However, distributed techniques are characterized by an intrinsically higher degree of complexity in the design phase with respect to centralized controllers. This could represent a great obstacle to their diffusion in the industrial world, and motivates the development of many innovative distributed model predictive control (MPC) algorithms for large-scale systems, see the survey papers [25,6] and the book [22], where the most recent and popular algorithms have been collected and described.

According to the classification of [25], a new non-iterative, non-cooperative approach based on neighbor-to-neighbor communication, called distributed predictive control (DPC), has been described in [15,14,2,4], where its convergence and stability properties have also been extensively analyzed. However, for a practical application of DPC, a number of issues concerning its realization and tuning have still to be solved and its performances must be assessed in realistic simulation scenarios. For these reasons, the aim of this paper is to consider and provide easy solutions to the main realization issues related to DPC (and similar distributed MPC methods), i.e. the use of a discretization method preserving the sparsity of the original continuous-time system, the computation of the required invariant sets, and the definition, both in the off-line and in the on-line phases, of the reference trajectories to be followed by the state and control variables. The proposed algorithms are then used for the realization and tuning of DPC applied to three realistic simulation problems. Specifically, the continuous-time models of the temperature dynamics in a simple building, of the level in a four tank system, and of a flotation process are controlled with DPC and the obtained performances are compared to those of a centralized model predictive control (cMPC) algorithm.

The paper is organized as follows: in Section 2 the DPC algorithm is summarized, while in Section 3.1 the discretization method, preserving the sparsity of the underlying continuous-time system, called mE-ZOH and originally presented in [7,8,12], is illustrated. Simplified procedures for computing the RPI sets are presented in Section 3.3. In Section 3.4, two techniques for the distributed design of the reference trajectories are proposed, while Section 4 contains the considered simulation examples. Some conclusions are drawn in Section 5.

* Corresponding author: Tel.: +39 02 23993599; fax: +39 02 23993599.
E-mail address: marcello.farina@polimi.it (M. Farina).

http://dx.doi.org/10.1016/j.jprocont.2014.02.016
0959-1524/© 2014 Elsevier Ltd. All rights reserved.

Notation. A matrix is Schur stable if all its eigenvalues lie in the interior of the unit circle. The short-hand $\mathbf{v} = (v_1, \ldots, v_s)$ denotes a column vector with s (not necessarily scalar) components $v_1, \ldots, v_s$. The symbol $\oplus$ denotes the Minkowski sum, namely $C = A \oplus B$ if and only if $C = \{c : c = a + b, \text{forall}\ a \in A,\ b \in B\}$, while $\oplus_{i=1}^{M} A_i = A_1 \oplus \ldots \oplus A_M$. The Pontryagin difference is defined using the symbol $\ominus$, i.e. $C = A \ominus B$ if and only if $C = \{c : c + b \in A, \text{forall}\ b \in B\}$. For a discrete-time signal s_t and $a, b \in \mathbb{N}$, $a \le b$, we denote $(s_a, s_{a+1}, \ldots, s_b)$ with $s_{[a:b]}$. Given a generic compact set $\mathcal{L}$, $\mathcal{H} = \text{box}(\mathcal{L})$ is the smallest hyperrectangle containing $\mathcal{L}$ with faces perpendicular to the cartesian axis. Finally, $\|\mathcal{L}\|_\infty = \max\limits_{l \in \mathcal{L}} \|l\|_\infty$.

2. The basic distributed predictive control algorithm

In this section, the distributed predictive control (DPC) algorithm first presented in [15] and further developed in [4] is briefly described. Let us assume that the system is constituted by M linear, discrete-time, non-overlapping subsystems, dynamically coupled through states and inputs, and subject to state and control constraints. For each subsystem $\mathcal{S}_i$, the dynamics is given by

$$\mathbf{x}_{k+1}^{i} = \mathbf{A}_{ii}\mathbf{x}_k^i + \mathbf{B}_{ii}\mathbf{u}_k^i + \sum_{j=1, j\neq i}^{M} \{\mathbf{A}_{ij}\mathbf{x}_k^j + \mathbf{B}_{ij}\mathbf{u}_k^j\} + \mathbf{d}_k^i \tag{1}$$

where $\mathbf{x}_k^i \in \mathcal{X}_i \subseteq \mathcal{R}^{n_i}$ and $\mathbf{u}_k^i \in \mathcal{U}_i \subseteq \mathcal{R}^{m_i}$ are the state and input vectors of the ith subsystem $\mathcal{S}_i$ ($i = 1, \ldots, M$), $\mathbf{d}_k^i \in \mathcal{D}_i \subset \mathcal{R}^{n_i}$ is an unknown bounded disturbance and the sets $\mathcal{X}_i, \mathcal{U}_i$ and $\mathcal{D}_i$ are convex neighborhoods of the origin. The subsystem $\mathcal{S}_j$ is said to be a *neighbor* of the subsystem $\mathcal{S}_i$ if and only if $\mathbf{A}_{ij} \neq 0$ and/or $\mathbf{B}_{ij} \neq 0$, i.e., if and only if the states x_j and/or inputs u_j of $\mathcal{S}_j$ influence the dynamics of $\mathcal{S}_i$. The symbol $\mathcal{N}_i$ denotes the set of neighbors of $\mathcal{S}_i$ (which excludes i).

Letting $\mathbf{x}_k = (\mathbf{x}_k^1, \ldots, \mathbf{x}_k^M)$, $\mathbf{u}_k = (\mathbf{u}_k^1, \ldots, \mathbf{u}_k^M)$ and $\mathbf{d}_k = (\mathbf{d}_k^1, \ldots, \mathbf{d}_k^M)$, the overall collective system can be written as

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k + \mathbf{d}_k \tag{2}$$

where the matrices $\mathbf{A}$ and $\mathbf{B}$ have block entries $\mathbf{A}_{ij}$ and $\mathbf{B}_{ij}$ respectively, $\mathbf{x} \in \mathcal{X} = \prod_{i=1}^M \mathcal{X}_i \subseteq \mathcal{R}^n$, $n = \sum_{i=1}^M n_i$, $\mathbf{u} \in \mathcal{U} = \prod_{i=1}^M \mathcal{U}_i \subseteq \mathcal{R}^m$, $m = \sum_{i=1}^M m_i$, $\mathbf{d} \in \mathcal{D} = \prod_{i=1}^M \mathcal{D}_i \subset \mathcal{R}^n$, and $\mathcal{X}$, $\mathcal{U}$ are convex by convexity of $\mathcal{X}_i$ and $\mathcal{U}_i$, respectively.

Remark 1. System 2 can be seen as the state-space representation of a discrete-time empirical model obtained from data through identification procedures, for instance by means of impulse or step response experiments, or it can be computed as the the linearization and discretization of a continuous-time first principle model. In the latter case, the discretization procedure must guarantee to maintain the sparsity of the original continuous-time model, i.e., the mutual influences among the subsystems. This issue is discussed in the following Section 3.1.

The following assumption on decentralized stabilizability is needed.

Assumption 1. There exists a block diagonal matrix $\mathbf{K}$=diag($\mathbf{K}_1$, ..., $\mathbf{K}_M$), with $\mathbf{K}_i \in R^{m_i \times n_i}$, $i = 1, \ldots, M$ such that: (i) $\mathbf{A} + \mathbf{B}\mathbf{K}$ is Schur, (ii) $\mathbf{F}_{ii} = (\mathbf{A}_{ii} + \mathbf{B}_{ii}\mathbf{K}_i)$ is Schur, $i = 1, \ldots, M$.

At any time instant k, each subsystem $\mathcal{S}_i$ transmits to its neighbors its future state and input reference trajectories (to be later specified) defined over the prediction horizon N, and called $\tilde{\mathrm{x}}_{k+\nu}^i$ and $\tilde{\mathrm{u}}_{k+\nu}^i$, $\nu = 0, \ldots, N-1$, respectively. These trajectories coincide with the *assumed trajectories* introduced in [10]. By adding suitable constraints to its MPC formulation, $\mathcal{S}_i$ is able to guarantee that, for all $k \geq 0$, its real trajectories lie in specified time invariant neighborhoods of $\tilde{\mathrm{x}}^i$ and $\tilde{\mathrm{u}}^i$, i.e., $\mathbf{x}_k^i \in \tilde{\mathrm{x}}_k^i \oplus \mathcal{E}_i$ and $\mathbf{u}_k^i \in \tilde{\mathrm{u}}_k^i \oplus \mathcal{E}_i^{\mathcal{U}}$, where $0 \in \mathcal{E}_i$ and $0 \in \mathcal{E}_i^{\mathcal{U}}$. In this way, the dynamics (1) of $\mathcal{S}_i$ can be written as

$$\mathbf{x}_{k+1}^i = \mathbf{A}_{ii}\mathbf{x}_k^i + \mathbf{B}_{ii}\mathbf{u}_k^i + \sum_{j\in\mathcal{N}_i}\{\mathbf{A}_{ij}\tilde{\mathrm{x}}_k^j + \mathbf{B}_{ij}\tilde{\mathrm{u}}_k^j\} + \mathbf{w}_k^i \tag{3}$$

where

$$\mathbf{w}_k^i = \sum_{j\in\mathcal{N}_i}\{\mathbf{A}_{ij}(\mathbf{x}_k^j - \tilde{\mathrm{x}}_k^j) + \mathbf{B}_{ij}(\mathbf{u}_k^j - \tilde{\mathrm{u}}_k^j)\} + \mathbf{d}_k^i \in \mathcal{W}_i$$

and $\mathcal{W}_i = \oplus_{j\in\mathcal{N}_i}\{\mathbf{A}_{ij}\mathcal{E}_j \oplus \mathbf{B}_{ij}\mathcal{E}_j^{\mathcal{U}}\} \oplus \mathcal{D}_i$.

Each subsystem, using the algorithm proposed in [20], solves a robust MPC problem considering that its dynamics is given by (3), where the term $\sum_{j\in\mathcal{N}_i}(\mathbf{A}_{ij}\tilde{\mathrm{x}}_{k+\nu}^j + \mathbf{B}_{ij}\tilde{\mathrm{u}}_{k+\nu}^j)$ represents an input known in advance over the prediction horizon $\nu = 0, \ldots, N-1$, to be suitably compensated, and $\mathbf{w}_k^i$ is a bounded disturbance to be rejected.

Similarly to [20], a nominal model of subsystem $\mathcal{S}_i$ is associated to Eq. (3)

$$\mathrm{x}_{k+1}^i = \mathbf{A}_{ii}\mathrm{x}_k^i + \mathbf{B}_{ii}\mathrm{u}_k^i + \sum_{j\in\mathcal{N}_i}\{\mathbf{A}_{ij}\tilde{\mathrm{x}}_k^j + \mathbf{B}_{ij}\tilde{\mathrm{u}}_k^j\} \tag{4}$$

while the control law to be used for $\mathcal{S}_i$ is

$$\mathbf{u}_k^i = \mathrm{u}_k^i + \mathbf{K}_i(\mathbf{x}_k^i - \mathrm{x}_k^i) \tag{5}$$

where $\mathbf{K}_i$ must be chosen to satisfy Assumption 1.

Letting $\mathbf{z}_k^i = \mathbf{x}_k^i - \mathrm{x}_k^i$, in view of (3)–(5) one has

$$\mathbf{z}_{k+1}^i = \mathbf{F}_{ii}\mathbf{z}_k^i + \mathbf{w}_k^i \tag{6}$$

where $\mathbf{w}_k^i \in \mathcal{W}_i$. Since $\mathcal{W}_i$ is bounded and $\mathbf{F}_{ii}$ is Schur, there exists a robust positively invariant (RPI) set $\mathcal{Z}_i$ for (6) such that, for all $\mathbf{z}_k^i \in \mathcal{Z}_i$, then $\mathbf{z}_{k+1}^i \in \mathcal{Z}_i$. Given $\mathcal{Z}_i$ define, if possible, two sets, neighborhoods of the origin, $\Delta\mathcal{E}_i$ and $\Delta\mathcal{U}_i$, $i = 1, \ldots, M$ such that $\Delta\mathcal{E}_i \oplus \mathcal{Z}_i \subseteq \mathcal{E}_i$ and $\Delta\mathcal{U}_i \oplus K_i\mathcal{Z}_i \subseteq \mathcal{E}_i^{\mathcal{U}}$, respectively.

At any time instant k each subsystem $\mathcal{S}_i$ solves the following i-DPC problem.

$$\min_{\mathrm{x}_k^i, \mathrm{u}^i_{[k:k+N-1]}} V_i^N = \sum_{\nu=0}^{N-1}(\|\mathrm{x}_{k+\nu}^i\|^2_{\mathbf{Q}_i^o} + \|\mathrm{u}^i_{k+\nu}\|^2_{\mathbf{R}_i^o}) + \|\mathrm{x}^i_{k+N}\|^2_{\mathbf{P}_i^o} \tag{7}$$

subject to (4),

$$\mathbf{x}_k^i - \mathrm{x}_k^i \in \mathcal{Z}_i \tag{8}$$

and, for $\nu = 0, \ldots, N-1$

$$\mathrm{x}^i_{k+\nu} - \tilde{\mathrm{x}}^i_{k+\nu} \in \Delta\mathcal{E}_i \tag{9}$$

$$\mathrm{u}^i_{k+\nu} - \tilde{\mathrm{u}}^i_{k+\nu} \in \Delta\mathcal{U}_i \tag{10}$$

$$\mathrm{x}^i_{k+\nu} \in \hat{\mathcal{X}}_i \subseteq \mathcal{X}_i \ominus \mathcal{Z}_i \tag{11}$$

$$\mathrm{u}^i_{k+\nu} \in \hat{\mathcal{U}}_i \subseteq \mathcal{U}_i \ominus \mathbf{K}_i\mathcal{Z}_i \tag{12}$$

and to the terminal constraint

$$\mathrm{x}^i_{k+N} \in \hat{\mathcal{X}}_i^F \tag{13}$$

The choice of the positive definite matrices $\mathbf{Q}_i^o$, $\mathbf{R}_i^o$, and $\mathbf{P}_i^o$ in (7) is discussed in Section 3.2 to guarantee stability and convergence, while $\hat{\mathcal{X}}_i^F$ in (13) is a nominal terminal set which must be chosen to satisfy the following assumption.

Assumption 2. Letting $\hat{\mathcal{X}} = \prod_{i=1}^M \hat{\mathcal{X}}_i, \hat{\mathcal{U}} = \prod_{i=1}^M \hat{\mathcal{U}}_i$ and $\hat{\mathcal{X}}^F = \prod_{i=1}^M \hat{\mathcal{X}}_i^F$, it holds that:

1 $\hat{\mathcal{X}}^F \subseteq \hat{\mathcal{X}}$ is an invariant set for $x_{k+1} = (\mathbf{A}+\mathbf{BK})x_k$;
2 $u = \mathbf{K}x \in \hat{\mathcal{U}}$ for any $x \in \hat{\mathcal{X}}^F$;
3 for all $x_k \in \hat{\mathcal{X}}^F$ and, for a given constant $\kappa > 0$,

$$\mathbf{V}^F(x_{k+1}) - \mathbf{V}^F(x_k) \le -(1+\kappa)\ell(x_k, \mathbf{K}x_k) \tag{14}$$

where $V^F(x) = \sum_{i=1}^M V_i^F(x^i) = \sum_{i=1}^M \|x^i\|^2_{\mathbf{P}_i^o}$ and $\ell(x, u) = \sum_{i=1}^M \ell_i(x^i, u^i) = \sum_{i=1}^M (\|x^i\|^2_{\mathbf{Q}_i^o} + \|u^i\|^2_{\mathbf{R}_i^o})$.

At time k, let the pair $x^i_{k|k}$, $u^i_{[k:k+N-1|k]}$ be the solution to the i-DPC problem and define by $u^i_{k|k}$ the input to the nominal system (4). Then, according to (5), the input to the subsystem (1) is

$$\mathbf{u}^i_k = u^i_{k|k} + \mathbf{K}_i(\mathbf{x}^i_k - x^i_{k|k}) \tag{15}$$

Denoting by $x^i_{k+\nu|k}$ the state trajectory of system (4) stemming from $x^i_{k|k}$ and $u^i_{[k:k+N-1|k]}$, at time k it is also possible to compute $x^i_{[k+N|k]}$ and $\mathbf{K}_i x^i_{k+N|k}$. In DPC, these values incrementally define the trajectories of the reference state and input variables to be used at the next time instant $k+1$, that is

$$\tilde{x}^i_{k+N} = x^i_{k+N|k}, \quad \tilde{u}^i_{k+N} = \mathbf{K}_i x^i_{k+N|k} \tag{16}$$

We underline that, in nominal operating conditions, the only information to be transmitted consists in the reference trajectories updated as in (16). More specifically, at time step k, subsystem $\mathcal{S}_i$ computes $\tilde{x}^i_{k+N}$ and $\tilde{u}^i_{k+N}$ according to (16) and transmits their values to all the subsystems having $\mathcal{S}_i$ as neighbor, allowing them to update the reference trajectories.

3. Realization issues and numerical algorithms

3.1. Block-wise discretization of large-scale structured systems

In many control applications, the model of the plant is developed in the continuous-time starting from physical laws, see for instance the examples reported in Section 4. In this framework, the sparse structure of the model clearly represents physical connections (such as mass or energy flows) between the subsystems, each one of them described by the linear (or linearized) model

$$\dot{x}^i_{c,t} = \mathbf{A}_{c,ii}\mathbf{x}^i_{c,t} + \mathbf{B}_{c,ii}\mathbf{u}^i_{c,t} + \sum_{j=1, j\neq i}^{M} \{\mathbf{A}_{c,ij}\mathbf{x}^j_{c,t} + \mathbf{B}_{c,ij}\mathbf{u}^j_{c,t}\} + \mathbf{d}^i_{c,t} \tag{17}$$

where t is the continuous time and the notation is coherent (mutatis mutandis) with the one adopted in (2). Unfortunately, the sparse zero-nonzero pattern of the system (zero-nonzero matrices $\mathbf{A}_{c,ij}$, $\mathbf{B}_{c,ij}$) is lost when the exact ZOH (Zero-Order-Hold), Backward Euler, or bilinear transformations are used, while it is preserved by the Forward Euler (FE) transformation. However, it is well known that with FE some important properties of the underlying continuous time system can be lost; for example stability is maintained only for very small sampling times, which can be inadvisable in many digital control applications.

In distributed and decentralized control techniques based on MPC, where discrete-time models are mainly utilized, the loss of sparsity can easily result in an increase of the controller complexity, see the DPC algorithm described in the previous section, or the methods presented in [5,13,15,24]. For these reasons, in order to improve the performance of FE and to maintain sparsity, a new discretization method called Mixed Euler ZOH (ME-ZOH) has been proposed in [7,8], and its properties have been studied in [12]. In synthesis, ME-ZOH allow one to compute the matrices of (2) starting from the continuous-time model (17) as follows:

$$\mathbf{A}_{ii}(h) = e^{\mathbf{A}_{c,ii}h} \tag{18}$$

$$\mathbf{A}_{ij}(h) = \int_0^h e^{\mathbf{A}_{c,ii}t}dt\ \mathbf{A}_{c,ij}, \quad j \neq i \tag{19}$$

$$\mathbf{B}_{ij}(h) = \int_0^h e^{\mathbf{A}_{c,ii}t}dt\ \mathbf{B}_{c,ij}, \quad \forall\ i, j \tag{20}$$

where h is the adopted sampling time. It is apparent that the zero-nonzero structure of the matrices $\mathbf{A}_{c,ij}$ and $\mathbf{B}_{c,ij}$ is maintained. This discretization method has been used in all the examples reported in Section 4.

3.2. Computation of the decentralized state-feedback gain

Algorithms for the design of the block-diagonal matrix $\mathbf{K}$ satisfying Assumption 1, and the computation of a positive-definite block-diagonal matrix $\mathbf{P} = \text{diag}(\mathbf{P}_1^o, \ldots, \mathbf{P}_M^o)$, $\mathbf{P}_i^o \in R^{n_i \times n_i}$ satisfying $(\mathbf{A}^T + \mathbf{K}^T\mathbf{B}^T)\mathbf{P}(\mathbf{A}+\mathbf{BK}) - \mathbf{P} \prec 0$ have been already presented in [4]. For completeness, one of the methods described in [4], based on the solution to suitable LMIs, is reported in the Appendix. This algorithm, which shares many similarities with the method recently proposed in [28], is less conservative with respect to the one originally proposed in [15], at the price of a larger computational burden.

To guarantee the convergence of the DPC algorithm, once $\mathbf{P}$ and $\mathbf{K}$ have been computed, parameters $\mathbf{Q}_i^o \succ 0$ and $\mathbf{R}_i^o \succ 0$ must be chosen such that $\mathbf{P} - (\mathbf{A}+\mathbf{BK})^T\mathbf{P}(\mathbf{A}+\mathbf{BK}) - (\mathbf{Q}+\mathbf{K}^T\mathbf{RK})(1+\kappa) \succ 0$, where $\mathbf{Q} = \text{diag}(\mathbf{Q}_1^o, \ldots, \mathbf{Q}_M^o)$, $\mathbf{R} = \text{diag}(\mathbf{R}_1^o, \ldots, \mathbf{R}_M^o)$ and κ is an arbitrary value greater than zero, see [15].

3.3. Simplified methods for the computation of sets

Two of the main issues in DPC are to verify, for all $i = 1, \ldots, M$ that (I) $\mathcal{E}_i \supseteq \mathcal{Z}_i \oplus \Delta\mathcal{E}_i$, $\mathcal{E}_i^{\mathcal{U}} \supseteq \mathbf{K}_i\mathcal{Z}_i \oplus \Delta\mathcal{U}_i$, $\mathcal{Z}_i \subset \mathcal{X}_i$ and $\mathbf{K}_i\mathcal{Z}_i \subset \mathcal{U}_i$, and (II) $\hat{\mathcal{X}}_i^F \subseteq \hat{\mathcal{X}}_i$ and $\mathbf{K}_i\hat{\mathcal{X}}_i^F \subseteq \hat{\mathcal{U}}_i$.

Concerning (I), recall that $\mathcal{Z}_i$ is the RPI set for Eq. (6) where the disturbance term $\mathbf{w}^i_k$ lies in the set $\mathcal{W}_i$. For this reason, the problem can not be tackled by considering each subsystem separately. In this section we propose two alternative solutions to (I).

Furthermore, to verify (II) we simply set $\hat{\mathcal{X}}_i^F = \alpha\mathcal{Z}_i$ for all $i = 1, \ldots, M$, for a sufficiently small $\alpha \in (0, 1)$. Finally, remark that an algorithm for obtaining a polytopic invariant outer approximation of the minimal RPI set is presented in [23].

The first technique is based on an empirical simplified “distributed reachability analysis” procedure, which has obtained remarkable results in several applications. With respect to the algorithm presented in [4], we will use rectangular sets (i.e., through the “box” operation) to greatly simplify the set-theoretical computations (e.g., the Minkowski sum), at the price of slightly more conservative results.

Algorithm 1. Computation of the RPI sets – method 1

(1) For all $i = 1, \ldots, M$, arbitrarily choose hyperrectangles $\Delta\mathcal{E}_i$ and $\Delta\mathcal{U}_i$.
(2) Initialize $\mathcal{Z}_i = \oplus_{j\in\mathcal{N}_i}\{\text{box}(\mathbf{A}_{ij}\Delta\mathcal{E}_j) \oplus \text{box}(\mathbf{B}_{ij}\Delta\mathcal{U}_j)\} \oplus \text{box}(\mathcal{D}_i)$ for all $i = 1, \ldots, M$.
(3) For all $i = 1, \ldots, M$, compute $\mathcal{Z}_i^+ = \text{box}(\mathbf{F}_{ii}\mathcal{Z}_i) \oplus \{\oplus_{j\in\mathcal{N}_i}\{\text{box}(\mathbf{A}_{ij}\mathcal{Z}_j) \oplus \text{box}(\mathbf{B}_{ij}\mathbf{K}_j\mathcal{Z}_j)\}\} \oplus \{\oplus_{j\in\mathcal{N}_i}\{\text{box}(\mathbf{A}_{ij}\Delta\mathcal{E}_j) \oplus \text{box}(\mathbf{B}_{ij}\Delta\mathcal{U}_j)\}\} \oplus \text{box}(\mathcal{D}_i)$.
(4) If $\mathcal{Z}_i^+ \subseteq \mathcal{Z}_i$ for all $i = 1, \ldots, M$ then go to step 5: by definition, the hyperrectangles $\mathcal{Z}_i$ actually correspond to the required RPI sets. Otherwise set $\mathcal{Z}_i = \mathcal{Z}_i^+$ and repeat step 3.

(5) If $\mathcal{Z}_i \subset \mathcal{X}_i$ and $\mathbf{K}_i\mathcal{Z}_i \subset \mathcal{U}_i$ then stop. Otherwise set $\Delta\mathcal{E}_i = \gamma\Delta\mathcal{E}_i$, $\Delta\mathcal{U}_i = \gamma\Delta\mathcal{U}_i$, with $\gamma \in (0, 1)$, and go to step 2.

A second possibility for computing the RPI sets $\mathcal{Z}_i$ resorts to solving a linear programming (LP) problem. For all $i = 1, \ldots, M$, we define sets $\mathcal{E}_i$, $\mathcal{E}_i^{\mathcal{U}}$, $\Delta\mathcal{E}_i$ and $\Delta\mathcal{U}_i$ as hypercubes, centered at the origin, with faces perpendicular to the cartesian axis and the scalars $e_i = \|\mathcal{E}_i\|_\infty$, $e_i^u = \|\mathcal{E}_i^{\mathcal{U}}\|_\infty$, $\Delta e_i = \|\Delta\mathcal{E}_i\|_\infty$, $\Delta u_i = \|\Delta\mathcal{U}_i\|_\infty$, and $d_i^\infty = \|\mathcal{D}_i\|_\infty$. Define also x_i^∞ and u_i^∞ as the infinity norms of the biggest cubes, centered at the origin, inscribed inside of $\mathcal{X}_i$ and $\mathcal{U}_i$, respectively. If we define $w_i^\infty = \sum_{j\in\mathcal{N}_i}\{\|\mathbf{A}_{ij}\|_\infty e_j + \|\mathbf{B}_{ij}\|_\infty e_j^u\} + d_i^\infty$, using the properties of norm operators it is possible to state that $w_i^\infty \geq \|\mathcal{W}_i\|_\infty$, where $\mathcal{W}_i$ is the set containing the real disturbance affecting subsystem i. To compute the RPI set $\mathcal{Z}_i$ for (6) (see [23]) we use the hypercube $\mathcal{W}_i^\infty$ having infinity norm w_i^∞, i.e., $\mathcal{Z}_i = \frac{1}{1-\alpha_i}\oplus_{l=0}^{s_i-1}\mathbf{F}_{ii}^l\mathcal{W}_i^\infty$, where s and $\alpha_i \in [0, 1)$ must fulfill $\mathbf{F}_{ii}^{s_i}\mathcal{W}_i^\infty \subseteq \alpha_i\mathcal{W}_i^\infty$. The latter is verified if $\|\mathbf{F}_{ii}\|_\infty^{s_i} \leq \alpha_i$ in view of properties of the norm operator and of the hypercubes. In addition, remark that $\mathcal{Z}_i$ is contained inside the hypercube having infinity norm $\gamma_i w_i^\infty$, where $\gamma_i = 1/(1-\alpha_i)\sum_{l=0}^{s_i-1}\|\mathbf{F}_{ii}\|_\infty^l$. These considerations suggest the following procedure for computing $\mathcal{Z}_i$.

Algorithm 2. Computation of the RPI sets – method 2

(1) For all $i = 1, \ldots, M$, arbitrarily choose parameters α_i.
(2) For all $i = 1, \ldots, M$, compute s_i such that $\|\mathbf{F}_{ii}\|_\infty^{s_i} \leq \alpha_i$ and than evaluate $\gamma_i = (1-\alpha_i)^{-1}\sum_{l=0}^{s_i-1}\|\mathbf{F}_{ii}\|_\infty^l$.
(3) Solve the following linear programming problem.

$$\min_{o_v} \rho \tag{21}$$

subject to

$$\rho \geq \gamma_i w_i^\infty \quad \forall i = 1, \ldots, M \tag{22}$$

$$\gamma_i w_i^\infty + \Delta e_i \leq e_i \quad \forall i = 1, \ldots, M \tag{23}$$

$$\|\mathbf{K}_i\|_\infty \gamma_i w_i^\infty + \Delta u_i \leq e_i^u \quad \forall i = 1, \ldots, M \tag{24}$$

$$\gamma_i w_i^\infty \leq x_i^\infty \quad \forall i = 1, \ldots, M \tag{25}$$

$$\|\mathbf{K}_i\|_\infty \gamma_i w_i^\infty \leq u_i^\infty \quad \forall i = 1, \ldots, M \tag{26}$$

$$\Delta e_i \geq \Delta\bar{e}_i \quad \forall i = 1, \ldots, M \tag{27}$$

$$\Delta u_i \geq \Delta\bar{u}_i \quad \forall i = 1, \ldots, M \tag{28}$$

where $o_v = (\Delta e_1, e_1, \Delta u_1, e_1^u, \ldots, \Delta e_M, e_M, \Delta u_M, e_M^u) \in \mathcal{R}^{4M}$ contains only strictly positive elements. $\Delta\bar{e}_i$ and $\Delta\bar{u}_i$ are arbitrary positive parameters to be used in order to have sets $\Delta\mathcal{E}_i$ and $\Delta\mathcal{U}_i$ bigger than a prescribed size.
(4) Compute $\mathcal{Z}_i = (1-\alpha_i)^{-1}\oplus_{l=0}^{s_i-1}\mathbf{F}_{ii}^l\mathcal{W}_i^\infty$.

In the proposed optimization problem, the objective function combined together with constraints (22) aims at minimizing the bigger RPI set. Constraints (25) and (26) guarantee the existence of sets $\hat{\mathcal{X}}_i$ and $\hat{\mathcal{U}}_i$ for all the subsystems. Lastly, constraints (23) and (24), if the LP problem turns out to be feasible, allow one to find the hypercubes $\mathcal{E}_i$, $\Delta\mathcal{E}_i$, $\mathcal{E}_i^{\mathcal{U}}$ and $\Delta\mathcal{U}_i$ such that $\mathcal{E}_i \supseteq \mathcal{Z}_i \oplus \Delta\mathcal{E}_i$ and $\mathcal{E}_i^{\mathcal{U}} \supseteq \mathbf{K}_i\mathcal{Z}_i \oplus \Delta\mathcal{U}_i$.

3.4. Distributed methods for the computation of reference trajectories

The DPC algorithm assumes that an initial feasible initial reference trajectory exists. This problem can be cast as a purely offline design problem.

On the other hand, disturbances of unexpected entity could occur during the ordinary system operation, alterating the system's condition (e.g., by producing constraint violation, such that, $\mathbf{x}_{k+1}^i - \mathrm{x}_{k+1|k}^i \notin \mathcal{Z}_i$) with possible serious consequences on the future solution (e.g., concerning feasibility) of the control problems. Once this condition is detected by a given system $\mathcal{S}_i$, it must be broadcast to all other subsystems through an event-based emergency iterative transmission, and an extra-ordinary reset operation requires the recalculation of new suitable state and output reference trajectories for all subsystems. The simplest solution consists (consistently with the approach suggested in [9]) in generating such trajectories using a centralized controller. This has the drawback that a centralized controller must be designed together with the distributed ones, and that it must be kept activated while the system is running in order to recover the proper functioning of the process if unpredicted external disturbances affect the plant. Obviously, this need of a centralized "hidden" supervisor greatly reduce the advantages of utilizing a distributed control scheme.

In this section we present two different trajectory generation methods, useful both for offline reference trajectory generation (i.e., performed at time $k = 0$) and for extra-ordinary reset operations, requiring number of iterative information exchanges between neighbors. The first method (i.e., Algorithm 4) is applicable in case $\mathbf{x} \in \hat{\mathcal{X}}^F$, while the second one (i.e., Algorithm 5) can be used in case $\mathbf{x} \notin \hat{\mathcal{X}}^F$.

Therefore, we first need a procedure to check whether $\mathbf{x}_k \in \hat{\mathcal{X}}^F$, i.e., that $\mathbf{x}_k^i \in \hat{\mathcal{X}}_i^F$ for all $i = 1, \ldots, M$.

To this purpose we define some useful notation: denote with $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ the connected, undirected communication graph supporting the distributed control architecture for system (2). $\mathcal{V}$ is the set of M nodes, each corresponding a subsystem, while $\mathcal{A}$ is the set of undirected arcs connecting the nodes (given two nodes $i, j \in \mathcal{V}$, there exists an undirected arc – of unitary length – $i \leftrightarrow j \in \mathcal{A}$ if and only if $j \in \mathcal{N}_i$ or $i \in \mathcal{N}_j$). We denote with $P_{\max}^s$ the longest among all the shortest paths linking all the possible pairs of nodes in $\mathcal{V}$. $P_{\max}^s$ can be computed, for instance, using the Floyd–Warshall algorithm [16]. $P_{\max}^s$ represent the maximum number of hops required sending information from a node to all other vertices. The following procedure has to be executed.

Algorithm 3. Algorithm for evaluating whether $\mathbf{x}_k \in \hat{\mathcal{X}}^F$

(1) For all $i = 1, \ldots, M$, initialize $\mu_i = 1$ if $\mathbf{x}_k^i \in \hat{\mathcal{X}}_i^F$ or $\mu_i = 0$ if $\mathbf{x}_k^i \notin \hat{\mathcal{X}}_i^F$. Set $\nu = 0$.
(2) Receive μ_j from all $j \in \mathcal{N}_i$ and from all $j : i \in \mathcal{N}_j$. Set $\nu = \nu + 1$.
(3) For all $i = 1, \ldots, M$, set $\mu_i = \min_{j:(i\leftrightarrow j)\in\mathcal{A}\bigcup\{i\}}(\mu_j)$. If $\nu < P_{\max}^s$ go to step 2. If $\nu = P_{\max}^s$ go to step 4.
(4) For all $i = 1, \ldots, M$, if $\mu_i = 1$, then controller i can conclude that $\mathbf{x}_k \in \hat{\mathcal{X}}^F$. Otherwise, it holds that $\mathbf{x}_k \notin \hat{\mathcal{X}}^F$.

Note that, after $P_{\max}^s$ iterations, it holds that $\mu_i = \mu_j$ for all $i, j = 1, \ldots, M$.

We now present the two distributed techniques for generating the trajectories $\tilde{\mathrm{x}}_{[k:k+N-1]}^i$ and $\tilde{\mathrm{u}}_{[k:k+N-1]}^i$ that each subsystem has to transmit to its neighbors. The first one, to be used when the whole state $\mathbf{x}_k$ is inside $\hat{\mathcal{X}}^F$, is based on the auxiliary control law, and guarantees to find a solution. It requires N transmissions of information from each subsystem to its neighbors. The second one, instead, is an optimization-based procedure which has been proved to be very effective when $\mathbf{x}_k \notin \hat{\mathcal{X}}^F$. The latter provides also the minimum prediction horizon length N such that a reference trajectory exists for all subsystems.

Algorithm 4. Computation of the reference trajectories – Method 1 ($\mathbf{x} \in \hat{\mathcal{X}}^F$)

(1) For all $i = 1, \ldots, M$, initialize $\tilde{\mathrm{x}}_k^i = \mathbf{x}_k^i$ and $\tilde{\mathrm{u}}_k^i = \mathbf{K}_i\tilde{\mathrm{x}}_k^i$.

(2) Receive $\tilde{x}^j_k$ and $\tilde{u}^j_k$ from the neighbors ($j \in \mathcal{N}_i$). If $N = 1$ stop. If $N \geq 2$, set $\nu = 0$ and then go to step 3.

(3) For all $i = 1, \ldots, M$, update the state reference trajectory as $\tilde{x}^i_{k+\nu+1} = \mathbf{A}_{ii}\tilde{x}^i_{k+\nu} + \mathbf{B}_{ii}\tilde{u}^i_{k+\nu} + \sum_{j\in\mathcal{N}_i}\{\mathbf{A}_{ij}\tilde{x}^j_{k+\nu} + \mathbf{B}_{ij}\tilde{u}^j_{k+\nu}\}$ and set $\tilde{u}^i_{k+\nu+1} = \mathbf{K}_i\tilde{x}^i_{k+\nu+1}$.

(4) Receive $\tilde{x}^j_{k+\nu+1}$ and $\tilde{u}^j_{k+\nu+1}$ from the neighbors ($j \in \mathcal{N}_i$). If $\nu = N - 1$ stop. Else, set $\nu = \nu + 1$ and go to step 3.

This first algorithm is very intuitive, because it exploits the properties of the invariant terminal set $\hat{\mathcal{X}}^F$. The N transmissions of information allow a distributed state evolution equal to the one that would result from applying, in a distributed fashion, the auxiliary control law to the entire system.

Algorithm 5. Computation of the reference trajectories – Method 2 ($\mathbf{x} \notin \hat{\mathcal{X}}^F$)

(1) For all $i = 1, \ldots, M$, set $\nu = 0$, define sets $\mathcal{B}_i$ containing the origin (their role will be later specified), initialize $\tilde{x}^i_k = \mathbf{x}^i_k$ and receive $\tilde{x}^j_k$ for all $j \in \mathcal{N}_i$ and for all j such that $\mathbf{B}_{ji} \neq 0$.

(2) For all $i = 1, \ldots, M$, initialize $\tilde{u}^i_{k-1}$ solving the following quadratic programming (QP) problem

$$\min_{\tilde{u}^i_{k-1}} \|\hat{\tilde{\mathbf{x}}}^{0ii}_{k+1}\|^2 + \sum_{j:\mathbf{B}_{ji}\neq 0} \frac{\|\mathbf{B}_{ji}\|_2^2}{\|\mathbf{B}_{ii}\|_2^2} \|\hat{\tilde{\mathbf{x}}}^{0ji}_{k+1}\|^2 \tag{29}$$

subject to

$$\hat{\tilde{\mathbf{x}}}^{0ii}_{k+1} = \mathbf{A}_{ii}\tilde{x}^i_k + \mathbf{B}_{ii}\tilde{u}^i_{k-1} + \sum_{j\in\mathcal{N}_i} \mathbf{A}_{ij}\tilde{x}^j_k \tag{30}$$

$$\hat{\tilde{\mathbf{x}}}^{0ji}_{k+1} = \mathbf{A}_{jj}\tilde{x}^j_k + \mathbf{B}_{ji}\tilde{u}^i_{k-1} + \mathbf{A}_{ji}\tilde{x}^i_k \tag{31}$$

$$\tilde{u}^i_{k-1} \in \hat{\mathcal{U}}_i \tag{32}$$

$$\hat{\tilde{\mathbf{x}}}^{ii}_{k+1} \in \hat{\mathcal{X}}_i \tag{33}$$

(3) For all $i = 1, \ldots, M$
- if $\nu = 0$ receive $\tilde{u}^j_{k-1}$ for all $j \in \mathcal{N}_i$;
- if $\nu \geq 1$ receive $\tilde{x}^j_{k+\nu}$ for all $j \in \mathcal{N}_i$.

(4) For all $i = 1, \ldots, M$, for all $j : \mathbf{B}_{ij} \neq 0$ compute $\lambda^{ij}_{k+\nu} = \mathbf{A}_{ii}\tilde{\mathbf{x}}^i_{k+\nu} + \mathbf{B}_{ii}\tilde{\mathbf{u}}^i_{k+\nu-1} + \sum_{z\in\mathcal{N}_i\setminus\{j\}}\{\mathbf{A}_{iz}\tilde{\mathbf{x}}^z_{k+\nu} + \mathbf{B}_{iz}\tilde{\mathbf{u}}^z_{k+\nu-1}\}$.

(5) For all $i = 1, \ldots, M$, for all $j : \mathbf{B}_{ji} \neq 0$, receive $\lambda^{ji}_{k+\nu}$.

(6) For all $i = 1, \ldots, M$, compute $\tilde{u}^i_{k+\nu}$ solving the following quadratic programming (QP) problem

$$\min_{\tilde{u}^i_{k+\nu}} \|\hat{\tilde{\mathbf{x}}}^{ii}_{k+\nu+1}\|^2 + \sum_{j:\mathbf{B}_{ji}\neq 0} \frac{\|\mathbf{B}_{ji}\|_2^2}{\|\mathbf{B}_{ii}\|_2^2} \|\hat{\tilde{\mathbf{x}}}^{ji}_{k+\nu+1}\|^2 \tag{34}$$

subject to

$$\hat{\tilde{\mathbf{x}}}^{ii}_{k+\nu+1} = \mathbf{A}_{ii}\tilde{x}^i_{k+\nu} + \mathbf{B}_{ii}\tilde{u}^i_{k+\nu} + \sum_{j\in\mathcal{N}_i}\{\mathbf{A}_{ij}\tilde{x}^j_{k+\nu} + \mathbf{B}_{ij}\tilde{u}^j_{k+\nu-1}\} \tag{35}$$

$$\hat{\tilde{\mathbf{x}}}^{ji}_{k+\nu+1} = \mathbf{A}_{ji}\tilde{x}^i_{k+\nu} + \mathbf{B}_{ji}\tilde{u}^i_{k+\nu} + \lambda^{ji}_{k+\nu} \tag{36}$$

$$\tilde{u}^i_{k+\nu} \in \hat{\mathcal{U}}_i \tag{37}$$

$$\tilde{u}^i_{k+\nu} - \tilde{u}^i_{k+\nu-1} \in \mathcal{B}_i \tag{38}$$

$$\hat{\tilde{\mathbf{x}}}^{ii}_{k+\nu+1} \in \hat{\mathcal{X}}_i \ominus \oplus_{j\in\mathcal{N}_i}\mathbf{B}_{ij}\mathcal{B}_j \tag{39}$$

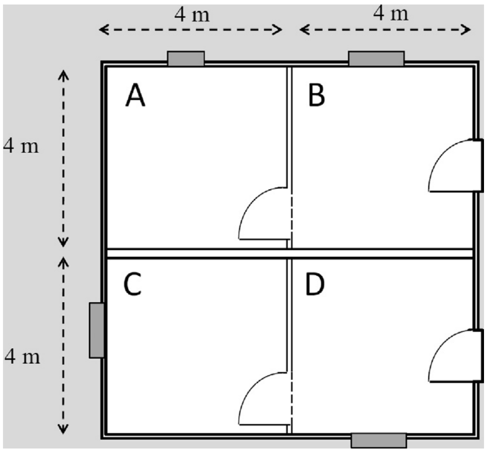

Fig. 1. Schematic representation of a building with two apartments.

(7) For all $i = 1, \ldots, M$, receive $\tilde{u}^j_{k+\nu}$ for all $j \in \mathcal{N}_i$.

(8) For all $i = 1, \ldots, M$, update the state reference trajectory as $\tilde{x}^i_{k+\nu+1} = \mathbf{A}_{ii}\tilde{x}^i_{k+\nu} + \mathbf{B}_{ii}\tilde{u}^i_{k+\nu} + \sum_{j\in\mathcal{N}_i}\{\mathbf{A}_{ij}\tilde{x}^j_{k+\nu} + \mathbf{B}_{ij}\tilde{u}^j_{k+\nu}\}$.

(9) If $\mathbf{x}_{k+\nu+1} \in \mathcal{X}^F_i$ for all $i = 1, \ldots, M$, then $N = \nu + 1$ and stop. Else, set $\nu = \nu + 1$ and go to step 3.

The second algorithm aims at iteratively finding feasible inputs using one-step predictions. Each controller i minimizes a cost function including both the norm of the state variable of subsystem i and the term $\frac{\|\mathbf{B}_{ji}\|_2^2}{\|\mathbf{B}_{ii}\|_2^2}\|\hat{\tilde{\mathbf{x}}}^{ji}_{k+\nu+1}\|^2$, limiting the possible negative effect of the inputs of subsystem i on the state of subsystem j, for all j such that $B_{ji} \neq 0$. The importance of this factor becomes greater as the coupling strength through inputs increases. The one-step prediction Eq. (35) is affected only by the errors on its neighbors' current inputs but, at the same time, such error is bounded using constraint (38). Finally note that the check of the stopping criterion in step 9) requires Algorithm 3 to be applied; to reduce the iterations required, one can check $\mathbf{x}_{k+\nu+1} \in \mathcal{X}^F$ only after a given number of iterations and, in case, only periodically.

4. Simulation examples

In this section, the DPC algorithm and the numerical methods presented in the previous sections are used in some simulation examples concerning popular case studies in the context of distributed control.

4.1. Temperature control

We aim at regulating the temperatures T_A, T_B, T_C and T_D of the four rooms of the building represented in Fig. 1 (see [11,3]). The first apartment is constituted by rooms A and B, while the second one by rooms C and D. Each room is equipped with a radiator supplying heats q_A, q_B, q_C and q_D. The heat transfer coefficient between rooms A–C and B–D is $k_1^t = 1\,\mathrm{W/m^2\,K}$, the one between rooms A–B and C–D is $k_2^t = 2.5\,\mathrm{W/m^2\,K}$, and the one between each room and the external environment is $k_e^t = 0.5\,\mathrm{W/m^2\,K}$. The nominal external temperature is $\overline{T}_E = 0\,^\circ\mathrm{C}$ and, for the sake of simplicity, solar radiation is not considered. The volume of each room is $V = 48\ \mathrm{m}^3$, and the wall surfaces between the rooms are all equal to $s_r = 12\ \mathrm{m}^2$, while those of the external walls are equal to $s_e = 24\ \mathrm{m}^2$. Air density and heat capacity are $\rho = 1.225\ \mathrm{kg/m^3}$ and $c = 1005\ \mathrm{J/kg\,K}$, respectively.

Letting $\phi=\rho cV$, the dynamic model is the following:

$$\phi\frac{dT_A}{dt}=s_rk_2^t(T_B-T_A)+s_rk_1^t(T_C-T_A)+s_ek_e^t(T_E-T_A)+q_A$$

$$\phi\frac{dT_B}{dt}=s_rk_2^t(T_A-T_B)+s_rk_1^t(T_D-T_B)+s_ek_e^t(T_E-T_B)+q_B$$

$$\phi\frac{dT_C}{dt}=s_rk_1^t(T_A-T_C)+s_rk_2^t(T_D-T_C)+s_ek_e^t(T_E-T_C)+q_C$$

$$\phi\frac{dT_D}{dt}=s_rk_1^t(T_B-T_D)+s_rk_2^t(T_C-T_D)+s_ek_e^t(T_E-T_D)+q_D$$

The considered equilibrium point is: $q_A=q_B=q_C=q_D=\overline{q}=\overline{T}s_ek_e^t$, with $T_A=T_B=T_C=T_D=\overline{T}=20\,^\circ\mathrm{C}$. Let $\delta T_A=T_A-\overline{T}$, $\delta T_B=T_B-\overline{T}$, $\delta T_C=T_C-\overline{T}$, $\delta T_D=T_D-\overline{T}$, $\delta T_E=T_E-\overline{T}_E$, $\delta q_A=(q_A-\overline{q})/c\rho V$, $\delta q_B=(q_B-\overline{q})/c\rho V$, $\delta q_C=(q_C-\overline{q})/c\rho V$ and $\delta q_D=(q_D-\overline{q})/c\rho V$. In this way, denoting $\sigma_1=s_rk_1^t/c\rho V$, $\sigma_2=s_rk_2^t/c\rho V$, $\sigma_3=s_ek_e^t/c\rho V$, $\sigma=\sigma_1+\sigma_2+\sigma_3$, $\mathbf{x}=(\delta T_A,\,\delta T_B,\,\delta T_C,\,\delta T_D)$, $\mathbf{u}=(\delta q_A,\,\delta q_B,\,\delta q_C,\,\delta q_D)$ and $\mathbf{d}=[\sigma_e\,\sigma_e\,\sigma_e\,\sigma_e]^T\delta T_E$ the previous model is rewritten in state space representation $\dot{\mathrm{x}}(t)=\mathrm{A_c}\,\mathbf{x}(t)+\mathrm{B_c}\mathbf{u}(t)+\mathbf{d}(t)$, where

$$\mathrm{A_c}=\begin{bmatrix}-\sigma & \sigma_2 & \sigma_1 & 0\\ \sigma_2 & -\sigma & 0 & \sigma_1\\ \sigma_1 & 0 & -\sigma & \sigma_2\\ 0 & \sigma_1 & \sigma_2 & -\sigma\end{bmatrix},\quad \mathrm{B_c}=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}$$

The discrete-time system of the form (2) (with $n=4$ and $m=4$) is obtained by mE-ZOH discretization with sampling time $h=10$. The partition of inputs and states is:

$$x^{[1]}=\begin{bmatrix}\delta T_A & \delta T_B\end{bmatrix}^T,\ u^{[1]}=\begin{bmatrix}\delta q_A & \delta q_B\end{bmatrix}^T$$

$$x^{[2]}=\begin{bmatrix}\delta T_C & \delta T_D\end{bmatrix}^T,\ u^{[2]}=\begin{bmatrix}\delta q_C & \delta q_D\end{bmatrix}^T$$

The constraints on the inputs and the states of the linearized system have been chosen as:

$$x_{\min}^{[1]}=\begin{bmatrix}-5 & -5\end{bmatrix}^T,\ x_{\max}^{[1]}=\begin{bmatrix}5 & 5\end{bmatrix}^T$$

$$x_{\min}^{[2]}=\begin{bmatrix}-5 & -5\end{bmatrix}^T,\ x_{\max}^{[2]}=\begin{bmatrix}5 & 5\end{bmatrix}^T$$

$$u_{\min}^{[1]}=\begin{bmatrix}-0.038 & -0.038\end{bmatrix}^T,\ u_{\max}^{[1]}=\begin{bmatrix}0.030 & 0.030\end{bmatrix}^T$$

$$u_{\min}^{[2]}=\begin{bmatrix}-0.038 & -0.038\end{bmatrix}^T,\ u_{\max}^{[2]}=\begin{bmatrix}0.030 & 0.030\end{bmatrix}^T$$

Matrices $\mathbf{K}_i$ and $\mathbf{P}_i$ representing a feasible solution to (44)–(47) are:

$$\mathbf{K}_1 = \mathbf{K}_2=\begin{bmatrix}-0.0986 & -0.0005\\ -0.0005 & -0,0986\end{bmatrix}$$

$$\mathbf{P}_1 = \mathbf{P}_2=\begin{bmatrix}2.17\cdot 10^6 & 1\\ 1 & 2.17\cdot 10^6\end{bmatrix}$$

The selected weighting matrices are $Q_1^o=Q_2^o=R_1^o=R_2^o=I_2$. Algorithm 1 for computing the RPI sets has been used, while the initial reference trajectories have been generated using Algorithm 5.

In the simulations reported below, the perturbed initial conditions for $\delta T_A=-3.2\,^\circ\mathrm{C}$, $\delta T_B=-2.58\,^\circ\mathrm{C}$, $\delta T_C=-1.12\,^\circ\mathrm{C}$, $\delta T_D=3.55\,^\circ\mathrm{C}$ have been set, the real external temperature has been assumed to randomly vary between $-10\,^\circ\mathrm{C}$ and $10\,^\circ\mathrm{C}$ and a sudden decrease of temperature T_A has been forced at $t=350\,s$, representing for instance to the opening of a door, to show the capability of Algorithm 4 to recover the reference trajectories.

The results of the simulations, performed using the continuous-time process model, are shown in Fig. 2, while the values of the input variables are depicted in Fig. 3. In both these figures a comparison between DPC and a centralized MPC (cMPC), with the same state and control weighting matrices is provided, showing only a small reduction of performances.

To quantitatively assess the performance deterioration of DPC with respect to cMPC, the following two indices have been considered

$$ISRE=\sum_{i=1}^{M}\int_0^{T_{end}}\sqrt{x(t)^{[i]\prime}x(t)^{[i]}}dt \tag{40}$$

$$J=\sum_{i=1}^{M}\sum_{k=0}^{N_{end}}x_k^{[i]\prime}Q_ix_k^{[i]}+u_k^{[i]\prime}R_iu_k^{[i]} \tag{41}$$

where T_{end} is the final time and N_{end} is the total number of discrete-time steps of the simulation experiment. The values of $ISRE$ and J corresponding to the state transients of Figs. 2 and 3 with DPC and cMPC are reported in Table 1.

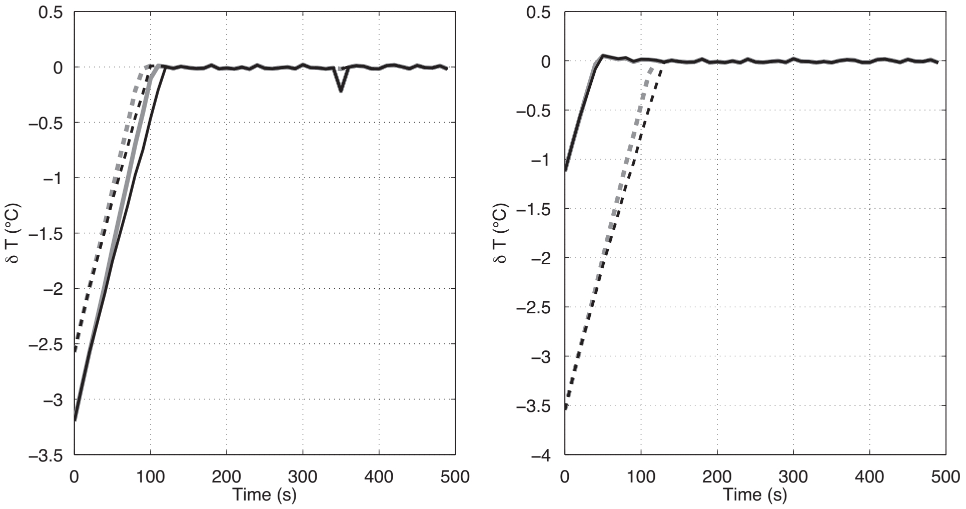

Fig. 2. State trajectories with DPC (black lines) and cMPC (gray lines) of δT_A (left, solid lines), δT_B (left, dashed lines), δT_C (right, solid lines), δT_D (right, dashed lines).

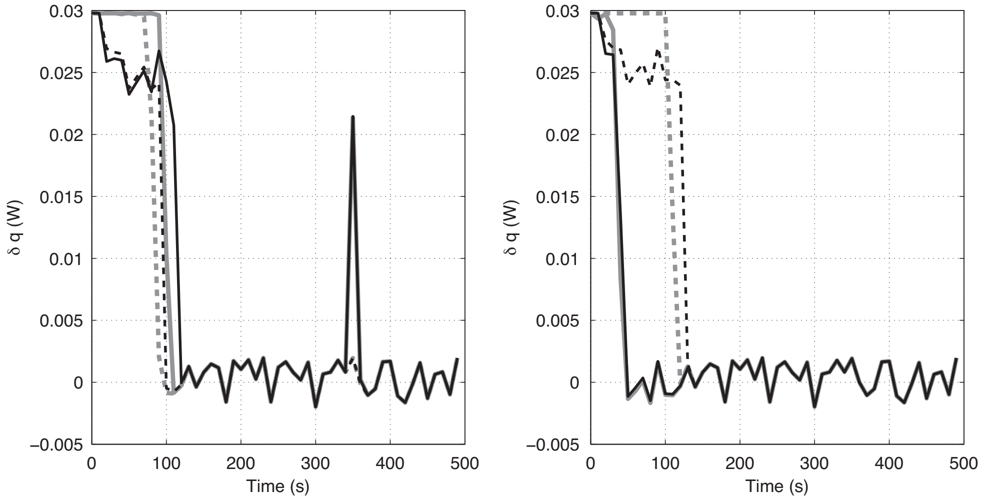

Fig. 3. Input trajectories with DPC (black lines) and cMPC (gray lines) of δq_A (left, solid lines), δq_B (left, dashed lines), δq_C (right, solid lines), δq_D (right, dashed lines).

Table 1
ISRE and *J* with DPC and cMPC in the temperature control problem.

ISRE	cMPC	461.4
	DPC	501.1
	DPC/cMPC	1.09
J	cMPC	120.2
	DPC	127.8
	DPC/cMPC	1.06

4.2. Four-tanks system

A benchmark case often used to assess the effectiveness of distributed control algorithms is the four-tanks system schematically drawn in Fig. 4, originally described in [17] and then utilized, for instance, in [1,21,2].

The goal is to regulate the levels h_1, h_2, h_3 and h_4 of the four tanks. The manipulated inputs are the voltages of the two pumps v_1 and v_2. We assume to have a bounded unknown disturbance $w = (w_1, w_2)$ on the applied voltages, such that the real input to the plant is $(v_1 + w_1, v_2 + w_2)$. Let the parameters γ_1 and $\gamma_2 \in (0, 1)$ represent the fraction of water that flows inside the lower tanks, and are kept fixed during the simulations. Then, the dynamics of the system is given by

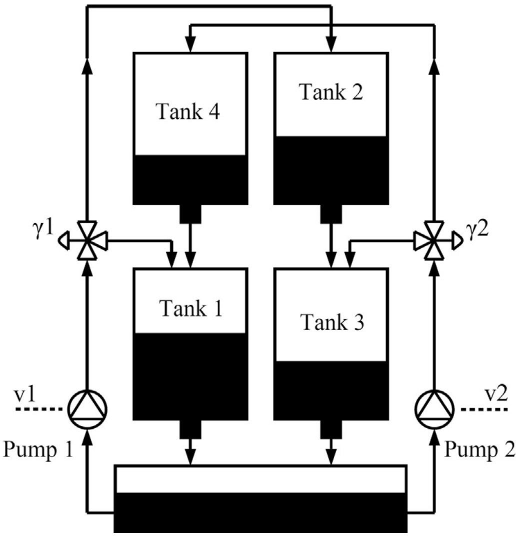

Fig. 4. Schematic representation of a four-tanks system.

$$
\begin{aligned}
\frac{dh_1}{dt} &= -\frac{a_1}{A_1}\sqrt{2gh_1} + \frac{a_4}{A_4}\sqrt{2gh_4} + \frac{\gamma_1 k_1}{A_1} v_1 \\
\frac{dh_2}{dt} &= -\frac{a_2}{A_2}\sqrt{2gh_2} + \frac{(1-\gamma_1)k_1}{A_2} v_1 \\
\frac{dh_3}{dt} &= -\frac{a_3}{A_3}\sqrt{2gh_3} + \frac{a_2}{A_2}\sqrt{2gh_2} + \frac{\gamma_2 k_2}{A_3} v_2 \\
\frac{dh_4}{dt} &= -\frac{a_4}{A_4}\sqrt{2gh_4} + \frac{(1-\gamma_2)k_2}{A_4} v_2
\end{aligned}
\tag{42}
$$

where A_i and a_i are the cross-section of Tank i and the cross section of the outlet hole of Tank i, respectively. The coefficients k_1 and k_2 represent the conversion parameters from the voltage applied to the pump to the flux of water. The values of the parameters, taken from [17], are: $A_1 = A_4 = 28$ cm^2, $A_2 = A_3 = 32$ cm^2, $a_1 = a_4 = 0.071$ cm^2, $a_2 = a_3 = 0.057$ cm^2, $k_1 = 3.35$ cm^3/*Vs*, $k_2 = 3.33$ cm^3/*Vs*, $\gamma_1 = 0.7$, $\gamma_2 = 0.6$. The considered equilibrium point is $\bar{v}_1 = \bar{v}_2 = 3$ V, $\bar{h}_1 = 12.263$ cm, $\bar{h}_2 = 1.409$ cm, $\bar{h}_3 = 12.783$ cm and $\bar{h}_4 = 1.634$ cm. Denoting $\delta h_l = h_l - \bar{h}_l$, $l = 1, 2, 3, 4$ and $\delta v_i = v_i - \bar{v}_i$, $i = 1, 2$, $\mathbf{x} = (\delta h_1, \delta h_2, \delta h_3, \delta h_4)$, $\mathbf{u} = (\delta v_1, \delta v_2)$, $\mathbf{d} = \mathbf{B}(w_1, w_2)$, linearizing system (42) around the considered equilibrium point and discretizing it using mE-ZOH with sampling time $h = 1$ s, we obtain a linear system of the type (2), where

$$
\mathbf{A} = \begin{bmatrix} 0.98 & 0 & 0 & 0.04 \\ 0 & 0.97 & 0 & 0 \\ 0 & 0.03 & 0.99 & 0 \\ 0 & 0 & 0 & 0.96 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0.08 & 0 \\ 0.03 & 0 \\ 0 & 0.06 \\ 0 & 0.05 \end{bmatrix}
$$

The inputs and states are partitioned as:

$$
x^{[1]} = \begin{bmatrix} \delta h_1 & \delta h_2 \end{bmatrix}^T, \; u^{[1]} = \delta v_1
$$

$$
x^{[2]} = \begin{bmatrix} \delta h_3 & \delta h_4 \end{bmatrix}^T, \; u^{[2]} = \delta v_2
$$

The constraints on the inputs and the states of the linearized system have been chosen as:

$$
x^{[1]}_{\min} = \begin{bmatrix} -12.263 & -1.409 \end{bmatrix}^T, \; x^{[1]}_{\max} = \begin{bmatrix} 40 & 40 \end{bmatrix}^T + x^{[1]}_{\min}
$$

$$
x^{[2]}_{\min} = \begin{bmatrix} -12.783 & -1.634 \end{bmatrix}^T, \; x^{[2]}_{\max} = \begin{bmatrix} 40 & 40 \end{bmatrix}^T + x^{[2]}_{\min}
$$

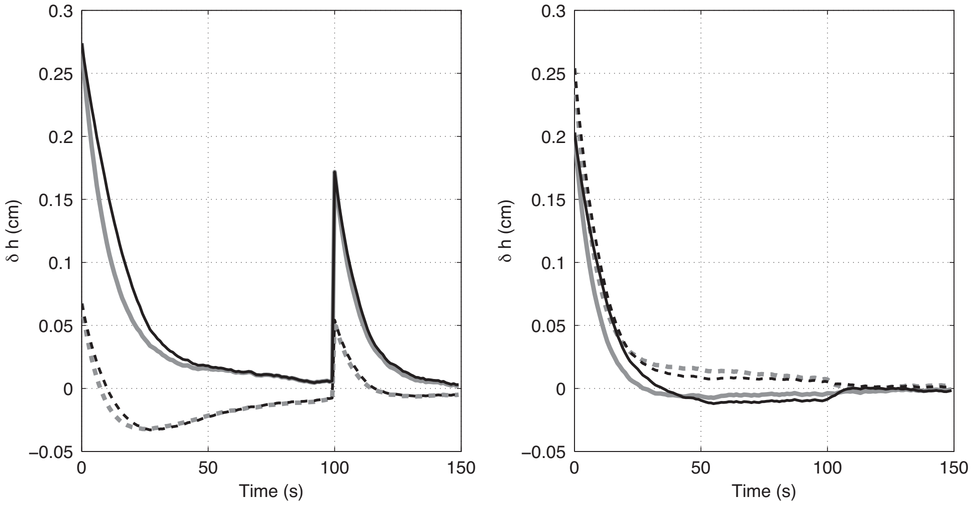

Fig. 5. Trajectories of the states $x^{[1]}$ (left) and $x^{[2]}$ (right) obtained with DPC (black lines) and with cMPC (gray lines) for the four-tanks system. Solid lines: δh_1 and δh_3; dashed lines: δh_2 and δh_4.

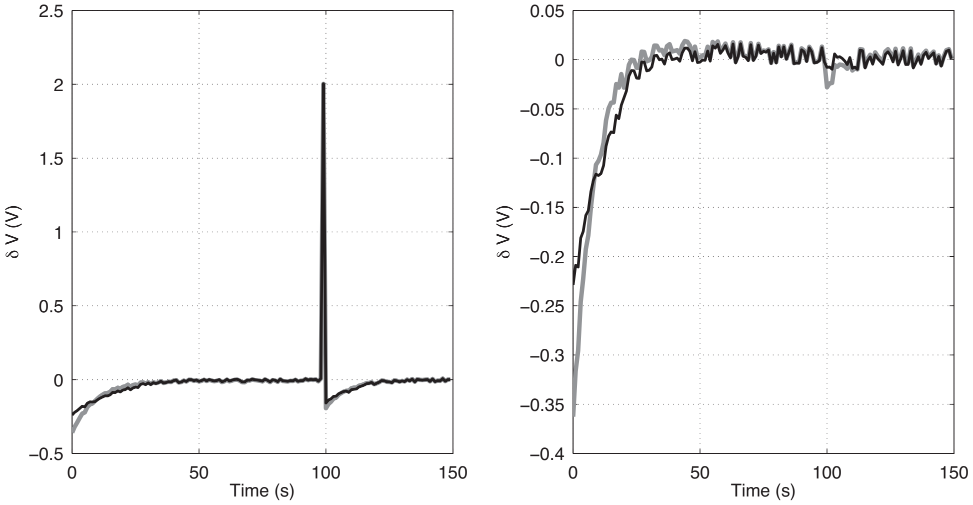

Fig. 6. Inputs δv_1 (left) and δv_2 (right) obtained with DPC (black lines) and with cMPC (gray lines) for the four-tanks system.

$$u_{\min}^{[1]} = u_{\min}^{[2]} - 3, \; u_{\max}^{[1]} = u_{\max}^{[2]} = 3$$

The disturbances $w_{1,2}$ on the applied voltages are assumed to randomly vary between −0.01 V and 0.01 V. Matrices $\mathbf{K}_i$ and $\mathbf{P}_i$ satisfying the LMI conditions are:

$$\begin{aligned} \mathbf{K}_1 &= \begin{bmatrix} -0.772 & -0.181 \end{bmatrix}, & \mathbf{K}_2 &= \begin{bmatrix} -0.778 & -0.250 \end{bmatrix} \\ \mathbf{P}_1 &= \begin{bmatrix} 48.3 & -1 \\ -1 & 59.3 \end{bmatrix}, & \mathbf{P}_2 &= \begin{bmatrix} 166.8 & 1.94 \\ 1.94 & 70.7 \end{bmatrix} \end{aligned}$$

The weighting matrices are $Q_1^o = Q_2^o = I_2$ and $R_1^o = R_2^o = 1$. To compute the RPI sets Algorithm 1 has been used, and the initial reference trajectories has been designed using Algorithm 4.

Starting from initial conditions $\delta h_1 = 0.274$ cm, $\delta h_2 = 0.067$ cm, $\delta h_3 = 0.203$ cm, and $\delta h_4 = 0.254$ cm. The simulation results, obtained using the continuous-time nonlinear model, are reported in Fig. 5, while in Fig. 6 the applied real voltages are shown. In addition to the external disturbance (w_1, w_2), included in the robust controller design, at time $t = 100$ s an unpredicted impulse equal to 2 V has been applied to the first pump. The reference trajectories were then re-generated online to recover the nominal operating conditions with Algorithm 4. The performances are close to the ones obtained with centralized MPC, as also witnessed by the values taken by the indices *ISRE* and *J* defined in (40) and (41) and reported in Table 2.

4.3. Cascade coupled flotation tanks

The third example deals with the level control problem of flotation tanks proposed in [27]. The system is constituted by five tanks connected in cascade with control valves between the tanks (Fig. 7). A flow of pulp q enters the first tank. The goal is to keep stable the levels y_i, $i = 1, \ldots, 5$, in all the tanks. The manipulated inputs are the commands to the valves v_i, $i = 1, \ldots, 5$.

Table 2
ISRE and *J* with DPC and cMPC in the in the four-tanks system problem.

ISRE	cMPC	74.2
	DPC	82.3
	DPC/cMPC	1.11
J	cMPC	1.36
	DPC	1.47
	DPC/cMPC	1.08

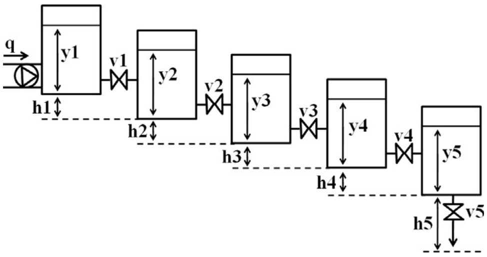

Fig. 7. Schematic representation of the flotation tanks.

The mathematical model describing the dynamics of the levels inside the five tanks is [27]:

$$
\begin{aligned}
\pi r^2 \frac{dy_1}{dt} &= q - k_1 v_1 \sqrt{y_1 - y_2 + h_1} \\
\pi r^2 \frac{dy_2}{dt} &= k_1 v_1 \sqrt{y_1 - y_2 + h_1} - k_2 v_2 \sqrt{y_2 - y_3 + h_2} \\
\pi r^2 \frac{dy_3}{dt} &= k_2 v_2 \sqrt{y_2 - y_3 + h_2} - k_3 v_3 \sqrt{y_3 - y_4 + h_3} \\
\pi r^2 \frac{dy_4}{dt} &= k_3 v_3 \sqrt{y_3 - y_4 + h_3} - k_4 v_4 \sqrt{y_4 - y_5 + h_4} \\
\pi r^2 \frac{dy_5}{dt} &= k_4 v_4 \sqrt{y_4 - y_5 + h_4} - k_5 v_5 \sqrt{y_5 + h_5}
\end{aligned}
\tag{43}
$$

where r is radius of the tanks, k_i, $i = 1, \ldots, 5$ are the valves coefficients and h_i, $i = 1, \ldots, 5$ are the physical height differences between subsequent tanks. We set $r = 1$ m, $k_i = 0.1$ $\mathrm{m}^{2.5}/\mathrm{Vs}$, $i = 1, \ldots, 5$ and $h_i = 0.5$ m, $i = 1, \ldots, 5$. The nominal value for the inlet flow is $\bar{q} = 0.1$ m^3/s and we assume it is affected by an uncertainty $w = \pm 0.5\%$ randomly varying with the time. We considered the equilibrium point where $\bar{y}_i = 2$ m, $i = 1, \ldots, 5$, and, correspondingly, $\bar{v}_i = 1.4142$ V, $i = 1, \ldots, 4$ and $\bar{v}_5 = 0.6325$ V. Let $\delta y_i = y_i - \bar{y}_i$, $i = 1, \ldots, 5$, $\delta v_i = v_i - \bar{v}_i$, $i = 1, \ldots, 5$, $\mathbf{x} = (\delta y_1, \delta y_2, \delta y_3, \delta y_4, \delta y_5)$, $\mathbf{u} = (\delta v_1, \delta v_2, \delta v_3, \delta v_4, \delta v_5)$ and $\mathbf{d} = \mathbf{B}_d w$. The linearization of system (43) in correspondence of the considered equilibrium point and its discretization with mE-ZOH using a sampling time 5 s, leads to a linear system of the form (2), where $\mathbf{B}_d = [1.4714\ 0\ 0\ 0\ 0]^T$ and

$$
\mathbf{A} = \begin{bmatrix}
0.853 & 0.147 & 0 & 0 & 0 \\
0.136 & 0.727 & 0.136 & 0 & 0 \\
0 & 0.136 & 0.727 & 0.136 & 0 \\
0 & 0 & 0.136 & 0.727 & 0.136 \\
0 & 0 & 0 & 0.157 & 0.969
\end{bmatrix},
$$

$$
\mathbf{B} = \begin{bmatrix}
-0.104 & 0 & 0 & 0 & 0 \\
0.096 & -0.096 & 0 & 0 & 0 \\
0 & 0.096 & -0.096 & 0 & 0 \\
0 & 0 & 0.096 & -0.096 & 0 \\
0 & 0 & 0 & 0.111 & -0.248
\end{bmatrix}
$$

The partitions of inputs and states, for $i = 1, \ldots, 5$ is:

$x^{[i]} = \delta y_i, \; u^{[1]} = \delta v_1$

The constraints on the inputs and the states of the linearized system, for $i = 1, \ldots, 5$, have been set as:

$x^{[i]}_{\mathrm{min}} = -1, \; x^{[i]}_{\mathrm{max}} = 1, \; u^{[i]}_{\mathrm{min}} = -\bar{v}_i, \; u^{[i]}_{\mathrm{max}} = 3 - \bar{v}_i$

Matrices $\mathbf{K}_i$ and $\mathbf{P}_i$ solving the LMI conditions are:

$$\mathbf{K}_1 = 0.287, \quad \mathbf{K}_2 = \mathbf{K}_3 = \mathbf{K}_4 = 0.143, \quad \mathbf{K}_5 = 0.776$$

$$\mathbf{P}_1 = 1.18, \quad \mathbf{P}_2 = 1.07, \quad \mathbf{P}_3 = 1.05, \quad \mathbf{O}_4 = 1, \quad \mathbf{K}_5 = 1$$

The weighting matrices, for $i = 1, \ldots, 5$, are $Q^o_i = R^o_i = 1$. To compute the RPI sets Algorithm 2 has been used, while the initial reference trajectories have been designed using Algorithm 5.

The initial levels of the tanks have been assumed to be different from the required values, that is $\delta y_1 = -23.3$ cm, $\delta y_2 = -21.6$ cm, $\delta y_3 = 23.3$ cm, $\delta y_4 = 44.4$ cm, and $\delta y_5 = -12.9$ cm and at time $t = 300$ s a disturbance of magnitude $w = 0.1$ m^3/s has been applied to the plant. In Fig. 8 we show the transients, obtained using the continuous-time nonlinear model, of the state and input of the first tank, directly affected by the external flow q. Figs. 9 and 10 report, respectively, the states and the inputs of the remaining four tanks. Note that, also in this case, the distributed control system reacts to the disturbance by generating from scratch the reference trajectories (with Algorithm 4). Moreover, only minor differences arise between the centralized and the distributed solutions, as again shown by the indices (40) and (41) reported in Table 3.

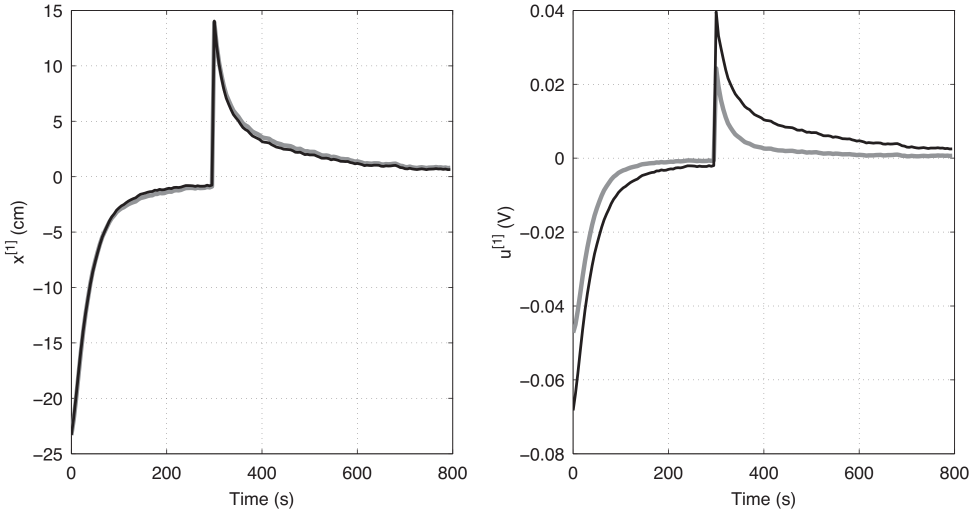

Fig. 8. Trajectories of the state $x^{[1]}$ (left) and of the input $u^{[1]}$ (right) obtained with DPC (black lines) and with cMPC (gray lines) for the control of the floating tanks.

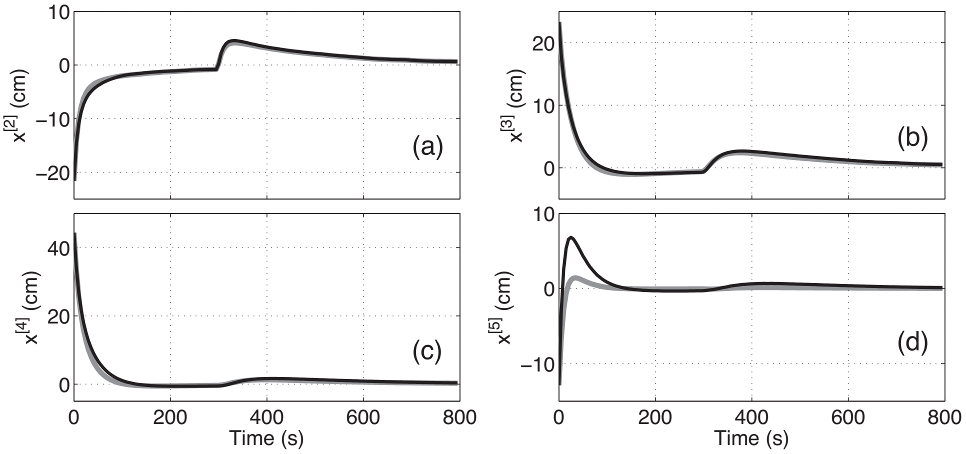

Fig. 9. Trajectories of the states $x^{[2]}$ (a), $x^{[3]}$ (b), $x^{[4]}$ (c) and $x^{[5]}$ (d) obtained with DPC (black lines) and with cMPC (gray lines) for the control of the floating tanks.

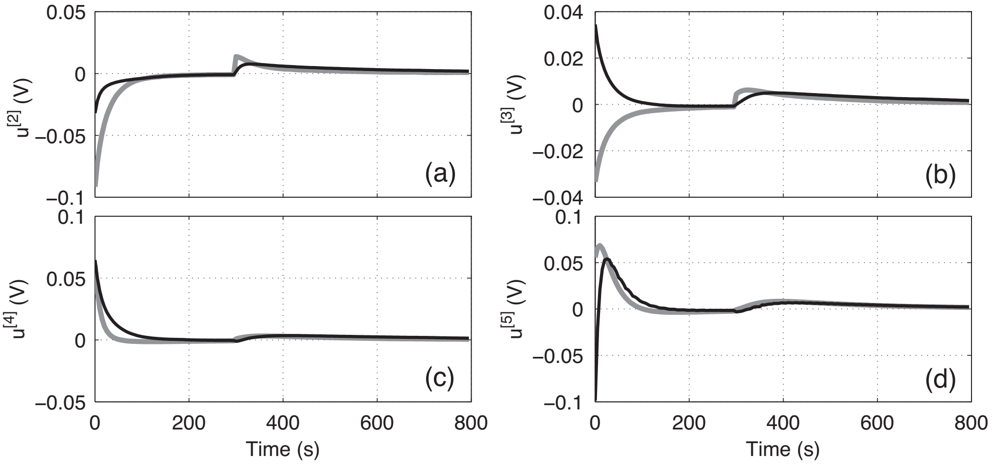

Fig. 10. Inputs $u^{[2]}$ (a), $u^{[3]}$ (b), $u^{[4]}$ (c) and $u^{[5]}$ (d) obtained with DPC (black lines) and with cMPC (gray lines) for the control of the floating tanks.

Table 3
ISRE and *J* with DPC and cMPC in the flotation tanks control problem.

ISRE	cMPC	10.7
	DPC	12.2
	DPC/cMPC	1.14
J	cMPC	6.6
	DPC	6.7
	DPC/cMPC	1.01

5. Conclusions

This paper has described some numerical algorithms and tuning rules for the realization of DPC, a non-iterative, non-cooperative approach based on neighbor-to-neighbor communication and characterized by a number of tuning parameters which must be carefully selected to guarantee stability and convergence properties. It is believed that these results can make DPC a practical solution to the design of innovative distributed predictive control algorithms, to be used in an industrial context.

The proposed algorithms, as well as the performances provided by DPC, have been tested in three examples. Notably, in these examples DPC has been used to control the nonlinear, continuous-time models of the considered systems, and the results achieved have been compared to those provided by a standard centralized MPC algorithm. Future works will consider the extension of the algorithm to enable plug and play capabilities.

Appendix A.

A.1. Computation of the feedback gain K

Define two matrices **S** and **Y** such that $\mathbf{K}=\mathbf{YS}^{-1}$ and $\mathbf{P}=\mathbf{S}^{-1}$, and consider the following set of LMI to be solved with respect to **Y** and **S**:

$$\begin{bmatrix} \mathbf{S} & \mathbf{SA}^T+\mathbf{Y}^T\mathbf{B}^T \\ \mathbf{AS}+\mathbf{BY} & \mathbf{S} \end{bmatrix} \succ 0 \tag{44}$$

$$\begin{bmatrix} \mathbf{S}_{ii} & \mathbf{S}_{ii}\mathbf{A}_{ii}^T+\mathbf{Y}_{ii}^T\mathbf{B}_{ii}^T \\ \mathbf{A}_{ii}\mathbf{S}_{ii}+\mathbf{B}_{ii}\mathbf{Y}_{ii} & \mathbf{S}_{ii} \end{bmatrix} \succ 0 \tag{45}$$

subject to

$$\mathbf{S}_{ij}=\mathbf{0} \quad \forall i,j=1,\ldots,M \quad (i\neq j) \tag{46}$$

$$\mathbf{Y}_{ij}=\mathbf{0} \quad \forall i,j=1,\ldots,M \quad (i\neq j) \tag{47}$$

where, for $i, j=1, \ldots, M$ $\mathbf{S}_{ij}\in\mathbb{R}^{(n_i+m_i)\times(n_j+m_j)}$, $\mathbf{Y}_{ij}\in\mathbb{R}^{m_i\times(n_j+m_j)}$ are the blocks entries of **S** and **Y**.

References

[1] I. Alvarado, D. Limon, D. Muñoz de la Peña, J.M. Maestre, M.A. Ridao, H. Scheu, W. Marquardt, R.R. Negenborn, B. De Schutter, F. Valencia, A comparative analysis of distributed MPC techniques applied to the HD-MPC four-tank benchmark, Journal of Process Control 21 (5) (2011) 800–815.

[2] G. Betti, M. Farina, R. Scattolini, Distributed predictive control for tracking constant references, in: Proceedings of the IEEE American Control Conference (ACC), 2012, pp. 6364–6369.

[3] G. Betti, M. Farina, R. Scattolini, Decentralized predictive control for tracking constant references, in: Proceedings of the 52th IEEE Conference on Decision and Control, 2013, pp. 5228–5233.

[4] G. Betti, M. Farina, R. Scattolini, Distributed predictive control: a non-cooperative approach based on robustness concepts, in: R. Negenborn, J. Maestre (Eds.), Distributed MPC Made Easy, Springer, 2013.

[5] E. Camponogara, D. Jia, B.H. Krogh, S. Talukdar, Distributed model predictive control, IEEE Control Systems Magazine 22 (1) (2002) 44–52.

[6] P.D. Christofides, R. Scattolini, D. Muñoz de la Peña, J. Liu, Distributed model predictive control: a tutorial review and future research directions, Computers and Chemical Engineering 51 (4) (2013) 21–41.

[7] P. Colaneri, M. Farina, S. Kirkland, R. Scattolini, R. Shorten, Positive systems: discretization with positivity and constraints, in: J. Daafouz, S. Tarbouriech, M. Sigalotti (Eds.), Hybrid Systems with Constraints, iSTE WILEY, 2013, pp. 1–20.

[8] P. Colaneri, M. Farina, R. Scattolini, R. Shorten, A note on discretization of sparce linear systems., in: Proceedings of the 7th IFAC Symposium on Robust Control Design, (ROCOND'12), 2012.

[9] W.B. Dunbar, Distributed receding horizon control of dynamically coupled nonlinear systems, IEEE Transactions on Automatic Control 52 (7) (2007) 1249–1263.

[10] W.B. Dunbar, R.M. Murray, Distributed receding horizon control of multi-vehicle formation stabilization, Automatica 42 (4) (2006) 549–558.

[11] M. Farina, G. Betti, R. Scattolini, A solution to the tracking problem using distributed predictive control, in: Proceedings of the 12th European Control Conference (ECC), 2013, pp. 4347–4352.

[12] M. Farina, P. Colaneri, R. Scattolini, Block-wise discretization accounting for structural constraints, Automatica 49 (11) (2013) 3411–3417.

[13] M. Farina, G. Ferrari-Trecate, R. Scattolini, Moving-horizon partition-based state estimation of large-scale systems, Automatica 46 (5) (2010) 910–918.

[14] M. Farina, R. Scattolini, An output feedback distributed predictive control algorithm, in: 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC), 2011, pp. 8139–8144.

[15] M. Farina, R. Scattolini, Distributed predictive control: a non-cooperative algorithm with neighbor-to-neighbor communication for linear systems, Automatica 48 (6) (2012) 1088–1096.

[16] Robert W. Floyd, Algorithm 97: shortest path, Communications of the ACM 5 (6) (1962) 345.

[17] K.H. Johansson, J.L.R. Nunes, A multivariable laboratory process with an adjustable zero, in: Proceedings of the IEEE American Control Conference, 1998, pp. 2045–2049.

[18] J. Lavaei, A. Momeni, A.G. Aghdam, A model predictive decentralized control scheme with reduced communication requirement for spacecraft formation, IEEE Transactions on Control Systems Technology 16 (2) (2008) 268–278.

[19] J. Lunze, Feedback Control of Large Scale Systems, Prentice Hall, 1992.

[20] D.Q. Mayne, M.M. Seron, S.V. Raković, Robust model predictive control of constrained linear systems with bounded disturbances, Automatica 41 (2) (2005) 219–224.

[21] M. Mercangöz, F.J. Doyle III, Distributed model predictive control of an experimental four-tank system, Journal of Process Control 17 (3) (2007) 297–308.

[22] R.R. Negenborn, J.M. Maestre (Eds.), Distributed Model Predictive Control Made Easy, vol. 310, Springer, 2014.

[23] S.V. Rakovic, E.C. Kerrigan, K.I. Kouramas, D.Q. Mayne, Invariant approximations of the minimal robust positively invariant set, IEEE Transactions on Automatic Control 50 (3) (2005) 406–410.

[24] S. Riverso, G. Ferrari-Trecate, Tube-based distributed control of linear constrained systems, Automatica 48 (11) (2012) 2860–2865.

[25] R. Scattolini, Architectures for distributed and hierarchical model predictive control-a review, Journal of Process Control 19 (5) (2009) 723–731.

[26] D.D. Siljak, Decentralized Control of Complex Systems, Academic Press, Cambridge, 1991.

[27] B. Stenlund, A. Medvedev, Level control of cascade coupled flotation tanks, Control Engineering Practice 10 (4) (2002) 443–448.

[28] P. Wang, B. Ding, Distributed receding horizon control for dynamically coupled large scale systems., in: 13th IFAC Symposium on Large Scale Complex Systems: Theory and Applications, 2013, pp. 254–259.