Cooperative control of dynamically decoupled systems via distributed model predictive control *

Matthias A. Müller, Marcus Reble, and Frank Allgöwer†

Abstract

In this paper, we propose a general framework for distributed model predictive control (DMPC) of discrete-time nonlinear systems with decoupled dynamics, but subject to coupled constraints and a common, cooperative task. In order to ensure recursive feasibility and convergence to the desired cooperative goal, the systems optimize a local cost function in a sequential order, while only neighbor-to-neighbor communication is allowed. In contrast to most of the existing DMPC schemes in the literature, we do not necessarily consider the stabilization of an a priori known setpoint. Instead, also other cooperative control tasks like consensus and synchronization problems can be handled within the proposed framework. In particular, one of our main contributions is to show how for the latter case the terminal cost functions and the terminal region can be suitably defined and computed. Furthermore, we illustrate our results with simulation examples.

1 Introduction

The problem of controlling networks of interacting dynamical systems has attracted a lot of attention in recent years. Besides classical control objectives like the stabilization of an a priori known setpoint, a variety of cooperative control tasks like consensus and synchronization between the systems are of great importance in such a context (see, e.g., [1–3] and the references therein). In particular, in consensus and synchronization problems, the systems have to agree on a common trajectory online, in contrast to following an a priori specified reference trajectory.

On the other hand, model predictive control (MPC) has become one of the most successful control strategies which finds application in many industrial processes. The major advantages are the possibility to explicitly take constraints into account and to optimize some performance criterion [4–6]. Considering the above, it is an interesting question how model predictive control can be used for decentralized and distributed control of networks of interacting systems. To this end, a variety of different settings and solution strategies have been proposed in the literature in recent years (see, e.g., [7–18] and the references therein). The various considered setups and

[*] A preliminary version of parts of this paper was presented at the IFAC World Congress 2011, Milano. This work was supported by the German Research Foundation (DFG) within the Priority Programme 1305 "Control Theory of Digitally Networked Dynamical Systems" and within the Cluster of Excellence in Simulation Technology (EXC 310/1) at the University of Stuttgart.

[†] The authors are with the Institute for Systems Theory and Automatic Control (IST), University of Stuttgart, Germany. e-mail: {matthias.mueller,marcus.reble,frank.allgower}@ist.uni-stuttgart.de

proposed different DMPC schemes can be classified in several ways, such as decentralized or noncooperative/cooperative distributed algorithms, coupled or uncoupled system dynamics, iterative or noniterative schemes, all-to-all or neighbor-to-neighbor communication, or according to the type of information exchanged between the systems (e.g. predicted state sequences or dual variables when using dual decomposition-based approaches), and so forth [7].

In this paper, the focus will be on dynamically decoupled systems which are coupled to each other through both coupling constraints as well as through the performance criterion, i.e., through a common objective. This setup is typically used when considering cooperative control tasks such as consensus and synchronization problems. The distributed MPC algorithms proposed in this paper are noniterative, and only neighbor-to-neighbor communication is required. Distributed MPC for the setting of dynamically decoupled systems was also considered in [8–11]. Stability was established via consistency constraints, meaning that at each time step, the newly calculated optimal trajectories of each system must not deviate too much from the ones calculated at the previous time step [8], or from the ones the neighboring systems assumed [9]. A different approach was taken in [10] for systems only coupled by constraints, where the systems solve their respective optimization problem sequentially, thus ensuring consistency over the network. In [11], the authors establish stability with an additional constraint requiring that the MPC control law asymptotically converges to some fixed local control law.

However, the vast majority of the existing DMPC algorithms in the literature, including those mentioned above, only consider the stabilization of an a priori known setpoint. There are only very few exceptions considering other cooperative tasks. In [19], consensus for single- and double-integrators is considered, where for the proof of convergence the optimal value function is not used as a Lyapunov function, but specific geometric properties of the system trajectories are utilized. In [20], for linear systems, the authors calculate an optimal consensus point at each time step by iteratively solving a centralized optimization problem, where this point is used as setpoint in the MPC formulation. A theoretical analysis of the proposed algorithm is given in [21], where convergence to a common consensus point corresponding to an equilibrium of the system is established.

The contribution of this paper is twofold. As a first main contribution (Section 3), we propose two distributed MPC algorithms where the goal is to asymptotically stabilize the overall closed-loop system with respect to some set $\mathcal{X}^0$. Herein, the first algorithm deals with the case of general cost functions, while in the second we exploit a certain separable structure of the cost functions. The proposed DMPC algorithms are rather general in nature and can be used for the solution of a number of distributed control problems. In particular, they are designed such that they also can be used for cooperative control tasks like consensus and synchronization problems. In the proposed algorithms, the systems optimize their performance criteria in a sequential order, similar to [10], where this idea was used for the robust stabilization of a setpoint for systems only coupled via constraints. The advantage of this approach is that less communication between the systems is needed in comparison to iterative schemes and that the control action is computed in a truly distributed way, in the sense that only neighboring systems have to communicate with each other and no centralized optimization problem is solved iteratively.

A second main contribution of the paper (Sections 4 and 5) is to explicitly show how the proposed DMPC algorithms of Section 3 can be used for two typical distributed control tasks. After applying the algorithms to the problem of stabilizing an a priori known setpoint and comparing our results to other existing DMPC schemes for this control task (Section 4), we in particular illustrate in Section 5 how the proposed algorithms can be used for the solution of consensus and synchronization problems of both linear and nonlinear systems. In contrast to [21], the common trajectory the systems agree upon does not necessarily have to be an equilibrium point of the system, which is crucial when e.g. considering the synchronization of oscillators. We specifically show how the terminal cost functions and the terminal region can be suitably defined and computed. Furthermore, we illustrate the applicability of our results with some simulation examples.

2 Preliminaries and setup

2.1 Notation

Denote by $\mathbb{R}$ the field of real numbers, and by $\mathbb{Z}_+$ the set of nonnegative integers. For any $a \in \mathbb{R}$, $|a|$ is the absolute value of a. For any vector $b_i \in \mathbb{R}^n$, define $\|b_i\|_2$ as the 2-norm, and $\|b_i\|_\infty$ as the ∞-norm of b_i. Let $\{b_i\}_{i\in\mathcal{I}}$ denote the collection of vectors b_i for all i in the index set $\mathcal{I}$. For a symmetric matrix $A \in \mathbb{R}^{n\times n}$, denote by $\lambda_{max}(A)$ its maximum eigenvalue. Let I_n and 0_n be the $n \times n$ identity and zero matrix, respectively. Let $A \otimes B \in \mathbb{R}^{np\times mq}$ be the Kronecker product of two matrices $A \in \mathbb{R}^{n\times m}$ and $B \in \mathbb{R}^{p\times q}$. $\Phi^+(x) := \Phi(f(x,u))$ for any function $\Phi(\cdot)$, when x satisfies $x(k+1) = f(x(k),u(k))$. For a set $\mathcal{A} \subseteq \mathbb{R}^n$, denote by $\text{int}(\mathcal{A})$ the interior of $\mathcal{A}$. For a set $\mathcal{A} \subseteq \mathbb{R}^n$ and a point $x \in \mathbb{R}^n$, the distance of x from the set $\mathcal{A}$ is defined as $|x|_\mathcal{A} := \inf_{z\in\mathcal{A}} |x-z|$. A function γ: $[0,\infty) \to [0,\infty)$ is of *class* $\mathcal{K}$ if γ is continuous, strictly increasing, and $\gamma(0) = 0$. If γ is also unbounded, it is of *class* $\mathcal{K}_\infty$. A function β: $[0,\infty) \times [0,\infty) \to [0,\infty)$ is of *class* $\mathcal{KL}$ if $\beta(\cdot,t)$ is of class $\mathcal{K}$ for each fixed $t \geq 0$, and $\beta(r,t)$ is decreasing in t with $\lim_{t\to\infty} = 0$ for each fixed $r \geq 0$.

2.2 Problem setup

Consider a network of N discrete-time dynamically decoupled systems s_i of the form

$$s_i: \qquad x_i(k+1) = f_i(x_i(k),u_i(k)), \qquad k \in \mathbb{Z}_+, \tag{1}$$

where $x_i \in \mathcal{X}_i \subseteq \mathbb{R}^{n_i}$ denotes the state and $u_i \in \mathcal{U}_i \subseteq \mathbb{R}^{m_i}$ the control input to system i, and $i \in \mathcal{I} := \{1,2,\ldots,N\}$. The input constraint sets $\mathcal{U}_i$ are assumed to be compact and to contain the origin in their interior. Denote by x and u the state and input of the overall system, i.e., $x := [x_1^T,\ldots,x_N^T]^T$ and $u := [u_1^T,\ldots,u_N^T]^T$, and let $\mathcal{X} := \mathcal{X}_1 \times \cdots \times \mathcal{X}_N$ and $\mathcal{U} := \mathcal{U}_1 \times \cdots \times \mathcal{U}_N$. Furthermore, denote by f the overall system dynamics such that $x(k+1) = f(x(k),u(k))$.

While the systems (1) are dynamically decoupled, they are coupled with each other via common constraints and a common objective (to be made precise later). Hence for any system i, we can define system j to be a neighbor of system i if the two systems are subject to coupled constraints or a coupled objective function. More formally, each of the systems s_i can be identified with a vertex of a graph $\mathcal{G} = (\mathcal{S},\mathcal{E})$, where $\mathcal{S} = \{s_1,...,s_N\}$ is the set of vertices, and the set of edges $\mathcal{E} \subseteq \{(s_i,s_j) \in \mathcal{S}\times\mathcal{S} | i \neq j\}$ describes the interconnection topology of the systems. Denote by $M := |\mathcal{E}|$ the number of edges of the graph $\mathcal{G}$. Let $\mathcal{N}_i := \{j|(s_i,s_j) \in \mathcal{E}\}$ be the set of indices of the neighbors of system i, and $d_i := |\mathcal{N}_i|$ its cardinality. According to the above definition of a neighbor, system i is a neighbor of system j if and only if also system j is a neighbor of system i, as coupling constraints and a coupled objective function affect both systems. This means that we model the graph $\mathcal{G}$ to be undirected, i.e., for any $i,j \in \mathcal{I}$, $(s_i,s_j) \in \mathcal{E}$ if and only if also $(s_j,s_i) \in \mathcal{E}$.

In order to be able to achieve the common objective and to satisfy the coupling constraints, in the following we assume that each system can communicate with each of its neighbors. In particular, this means that the systems can exchange information about their predicted trajectories with their neighbors, which we will make preciser later.

Our goal is to distributedly compute a control law such that the overall closed-loop system is asymptotically stable with respect to some closed set $\mathcal{X}^0 \subseteq \mathcal{X}$. Later on, we will consider as two special cases the situations where $\mathcal{X}^0$ is (*i*) a prespecified setpoint, and (*ii*) where $\mathcal{X}^0$ is the consensus subspace $\mathcal{X}^0 = \{x \in \mathcal{X} : x_1 = x_2 = \cdots = x_N\}$. We want to achieve the stabilization of $\mathcal{X}^0$ by DMPC, i.e., by minimizing at each time step a local performance criterion for each system, satisfying both local constraints as well as the constraints by which the systems are coupled with their neighbors. To this end, consider the following finite horizon open-loop optimal control problem for system i with prediction horizon T:

Problem $\mathcal{P}_i$: At time instant k,

$$\begin{aligned}
&\underset{\mathbf{u}_i(k)}{\text{minimize}}\; J_i(x_i(k), \tilde{\mathbf{x}}_{-i}(k), \mathbf{u}_i(k)) \\
&= \sum_{l=0}^{T-1} L_i\big(x_i(k+l|k), \tilde{x}_{-i}(k+l|k), u_i(k+l|k)\big) \\
&\quad + F_i\big(x_i(k+T|k), \tilde{x}_{-i}(k+T|k)\big)
\end{aligned} \tag{2a}$$

subject to

$$x_i(k+l+1|k) = f_i\big(x_i(k+l|k), u_i(k+l|k)\big) \tag{2b}$$

$$x_i(k|k) = x_i(k) \tag{2c}$$

$$u_i(k+l|k) \in \mathcal{U}_i \tag{2d}$$

$$x_i(k+l+1|k) \in \mathcal{X}_i \tag{2e}$$

$$z_q(x_i(k+l+1|k), \tilde{x}_{-i}(k+l+1|k)) \in \mathcal{Z}_q \tag{2f}$$

for all $l \in \{0, 1, \ldots, T-1\}$.

Herein, $\mathbf{u}_i(k)$ consists of the sequence of input values $\mathbf{u}_i(k) := \{u_i(k|k), \ldots, u_i(k+T-1|k)\}$ predicted at time k, $\mathbf{x}_i(k) := \{x_i(k|k), \ldots, x_i(k+T|k)\}$ is the corresponding predicted state sequence, and $\tilde{\mathbf{x}}_{-i}(k)$ consists of the sequences of planned state trajectories of system i's neighbors, i.e.,

$$\begin{aligned}
\tilde{\mathbf{x}}_{-i}(k) &= \{\tilde{x}_{-i}(k|k), \ldots, \tilde{x}_{-i}(k+T|k)\} \\
&:= \begin{bmatrix} \tilde{\mathbf{x}}_{i_1}(k) \\ \vdots \\ \tilde{\mathbf{x}}_{i_{d_i}}(k) \end{bmatrix} = \begin{bmatrix} \{\tilde{x}_{i_1}(k|k), \ldots, \tilde{x}_{i_1}(k+T|k)\} \\ \vdots \\ \{\tilde{x}_{i_{d_i}}(k|k), \ldots, \tilde{x}_{i_{d_i}}(k+T|k)\} \end{bmatrix},
\end{aligned} \tag{3}$$

where $\{i_1, \ldots, i_{d_i}\}$ is an ordered sequence of the elements of the set $\mathcal{N}_i$, i.e. $i_1 < \cdots < i_{d_i}$. As system i optimizes only over its local input trajectory $\mathbf{u_i}$, the planned trajectories of the neighbors, $\tilde{\mathbf{x}}_{-i}$, are held as constant parameters. We will specify later what $\tilde{\mathbf{x}}_{-i}$ exactly is, i.e., which trajectories system i assumes for its neighbors. The functions z_q and the sets $\mathcal{Z}_q \subseteq \mathbb{R}^{\ell_q}$ with $q \in \{1, ..., Q\}$

are the Q coupling outputs and coupling constraint sets the systems are confined to. Clearly, for Problem $\mathcal{P}_i$, only those of the Q coupling constraints are relevant where x_i appears explicitly. Both the stage cost L_i and the terminal cost F_i, which can both also depend on the neighboring states $\{x_j\}_{j\in\mathcal{N}_i}$, are assumed to be such that their respective sum is positive definite with respect to the set $\mathcal{X}^0$, i.e., $\sum_{i=1}^N L_i(x,u) \geq \alpha_1(|x|_{\mathcal{X}^0})$ for all $u \in \mathcal{U}$ and $\sum_{i=1}^N F_i(x) \geq \alpha_2(|x|_{\mathcal{X}^0})$, respectively, for some $\alpha_1, \alpha_2 \in \mathcal{K}_\infty$.

When using a terminal cost and terminal region approach in MPC, the crucial assumption in order to establish stability is that the terminal region can be made invariant under a local control law, and that the terminal cost function can be used as a control Lyapunov function inside the terminal region [5, 6]. We will make a similar assumption here.

Assumption 1 *There exists a terminal region $\mathcal{X}^f \subseteq \mathcal{X}$, and for each system $i \in \mathcal{I}$ an auxiliary local control law $u_i = k_i^{loc}(x_i, \{x_j\}_{j\in\mathcal{N}_i})$, such that the terminal region $\mathcal{X}^f$ is invariant with respect to the overall closed-loop system $x(k+1) = f(x(k), k^{loc}(x(k)))$ with $k^{loc} := [(k_1^{loc})^T, \dots, (k_n^{loc})^T]^T$, and the following holds for all $x \in \mathcal{X}^f$ and for all $i \in \mathcal{I}$:*

$$k_i^{loc}(x_i, \{x_j\}_{j\in\mathcal{N}_i}) \in \mathcal{U}_i \tag{4a}$$

$$z_q(x_i, \{x_j\}_{j\in\mathcal{N}_i}) \in \mathcal{Z}_q \qquad \forall q \in \{1, \dots, Q\} \tag{4b}$$

$$\sum_{i=1}^N F_i^+\big(x_i, \{x_j\}_{j\in\mathcal{N}_i}\big) - F_i\big(x_i, \{x_j\}_{j\in\mathcal{N}_i}\big) + L_i\big(x_i, \{x_j\}_{j\in\mathcal{N}_i}, k_i^{loc}(x_i, \{x_j\}_{j\in\mathcal{N}_i})\big) \leq 0 \tag{4c}$$

In Assumption 1, (4a) ensures the satisfaction of the input constraints when applying the auxiliary local control laws, (4b) the satisfaction of the coupling constraints inside the terminal region, and (4c) the decayrate of the sum of the terminal cost functions. The invariance condition for $\mathcal{X}^f$ is implied by (4c), if $\mathcal{X}^f$ is chosen as a sublevel set of the sum of the terminal cost functions. If not stated otherwise, in the remainder of this paper we will follow this approach, i.e., take

$$\mathcal{X}^f := \{x : \sum_{i=1}^N F_i\big(x_i, \{x_j\}_{j\in\mathcal{N}_i}\big) \leq \alpha\} \tag{5}$$

for some $\alpha > 0$ such that $\mathcal{X}^f \subseteq \mathcal{X}$. Note that the local controllers are only allowed to depend on neighboring system states, whereas we assume the existence of a "centralized" terminal region $\mathcal{X}^f$ for the overall system. We use such a centralized terminal region as in typical distributed control tasks such as consensus problems, only the decaying of the sum of the terminal cost functions (4c) can be ensured, but not the decaying of the single terminal cost functions. Hence only invariance of a centralized terminal region can be ensured, but not invariance of some decoupled terminal regions. In order to guarantee recursive feasibility and asymptotic stability of the overall closed-loop system with respect to the set $\mathcal{X}^0$ later on, we have to require that the overall system state at the end of the prediction horizon lies inside the terminal region, i.e.,

$$x(k+T|k) \in \mathcal{X}^f. \tag{6}$$

Note that this centralized terminal constraint cannot directly be included into Problem $\mathcal{P}_i$ as an additional terminal constraint, as system i does not have any information about non-neighboring systems. In Section 3, we will show how the centralized terminal constraint (6) can be satisfied nevertheless and dealt with in a distributed fashion. Furthermore, we show that in some cases it is possible to separate the centralized terminal constraint (6) in order to be able to include it directly into the optimization problems $\mathcal{P}_i$.

3 Distributed MPC algorithms for cooperative control

In this section, we will formulate and analyze two DMPC algorithms. As stated in the previous section, the goal is to stabilize the overall closed-loop system with respect to some closed set $\mathcal{X}^0$. The first algorithm considers the situation where no further assumptions are imposed on the type of stage and terminal cost functions L_i and F_i used in Problem $\mathcal{P}_i$. In the second algorithm, we consider stage and terminal cost functions L_i and F_i with a special structure which can be exploited in order to significantly simplify the proposed DMPC algorithm.

3.1 DMPC: general cost functions

The first proposed DMPC algorithm is specified as follows.

Algorithm 1 Distributed MPC: general cost functions.

0) Initialization: Set $k = 0$, and for all systems i, find a feasible solution $\hat{\mathbf{u}}_i(0)$ with corresponding state sequence $\hat{\mathbf{x}}_i(0)$, such that the constraints (2b)–(2f) and (6) are satisfied. Each system transmits $\hat{\mathbf{x}}_i(0)$ to its neighbors. Go to Step 2).

1) At time instant k, each system i computes the candidate input sequence

$$\hat{\mathbf{u}}_i(k) := \Big\{u_i^*(k|k-1), \ldots, u_i^*(k+T-1|k-1), k_i^{loc}\big(x_i^*(k+T-1|k-1), \tilde{x}_{-i}(k+T-1|k-1)\big)\Big\} \quad (7)$$

by taking the remaining part of the old optimal input sequence u_i^* and adding the auxiliary local control law k_i^{loc}, as well as the corresponding candidate state trajectory $\hat{\mathbf{x}}_i(k)$. Each system sends $\hat{\mathbf{x}}_i(k)$ to all of its neighbors.

2) After receiving the trajectories $\hat{\mathbf{x}}_j(k)$ from all of its neighbors $j \in \mathcal{N}_i$, each system i stacks them together into

$$\tilde{\mathbf{x}}_{-i}(k) = \Big[\hat{\mathbf{x}}_{i_1}(k)^T \ldots \hat{\mathbf{x}}_{i_{d_i}}(k)^T\Big]^T =: \hat{\mathbf{x}}_{-i}(k) \quad (8)$$

and computes $J_{i,old} := J_i(x_i(k), \tilde{\mathbf{x}}_{-i}(k), \hat{\mathbf{u}}_i(k))$ as well as $F_{i,old} := F_i(\hat{x}_i(k+T|k), \tilde{x}_{-i}(k+T|k))$.

3) For $i = 1$ to N, system i

   A) solves Problem $\mathcal{P}_i$, and denotes the solution by $\mathbf{u}_{i,test}^*(k)$, the corresponding state trajectory by $\mathbf{x}_{i,test}^*(k)$, the optimal value of the cost function by $J_{i,test}$, and the value of the corresponding terminal cost function by $F_{i,test}$.

   B) sends $\mathbf{x}_{i,test}^*(k)$ to its neighbors, which compute $J_{j,test} := J_j(x_j(k), \tilde{\mathbf{x}}_{-j,test}(k), \bar{\mathbf{u}}_j(k))$ and $F_{j,test} := F_j(\bar{x}_j(k+T|k), \tilde{x}_{-j,test}(k+T|k))$. Herein, $\tilde{\mathbf{x}}_{-j,test} := \tilde{\mathbf{x}}_{-j}$, but with $\hat{\mathbf{x}}_i(k)$ replaced by $\mathbf{x}_{i,test}^*(k)$. Furthermore, $\bar{\mathbf{u}}_j(k) = \mathbf{u}_j^*(k)$ and $\bar{\mathbf{x}}_j(k) = \mathbf{x}_j^*(k)$ if $j < i$, i.e., for all neighbors which already have calculated their optimal input trajectory at time k; otherwise, $\bar{\mathbf{u}}_j(k) = \hat{\mathbf{u}}_j(k)$ and $\bar{\mathbf{x}}_j(k) = \hat{\mathbf{x}}_j(k)$, i.e., for all neighbors which have not yet calculated their optimal input trajectory at time k.

C) receives $\delta_j := J_{j,test} - J_{j,old}$ as well as $e_j := F_{j,test} - F_{j,old}$ from all of its neighbors and checks whether

$$\delta_i + \sum_{j \in \mathcal{N}_i} \delta_j \leq 0, \tag{9}$$

$$e_i + \sum_{j \in \mathcal{N}_i} e_j \leq 0. \tag{10}$$

If (9)–(10) hold, set $\mathbf{u}_i^*(k) := \mathbf{u}_{i,test}^*(k)$, $\mathbf{x}_i^*(k) := \mathbf{x}_{i,test}^*(k)$, $J_{i,old} := J_{i,test}$ and $flag := 1$; otherwise, set $\mathbf{u}_i^*(k) := \hat{\mathbf{u}}_i(k)$, $\mathbf{x}_i^*(k) := \hat{\mathbf{x}}_i(k)$ and $flag := 0$.

D) sends $flag$ to its neighbors $j \in \mathcal{N}_i$. If $flag = 1$, the neighbors set $\tilde{\mathbf{x}}_{-j}(k) := \tilde{\mathbf{x}}_{-j,test}(k)$, $J_{j,old} := J_{j,test}$, and $F_{j,old} := F_{j,test}$.

4) Each system applies $k_i^{MPC} := u_i^*(k|k)$.

5) Increment k and go to Step 1).

In Steps 3B) and 3C) of Algorithm 1, it is checked whether the input trajectory newly calculated by optimizing the local performance criterion J_i leads to an overall decrease in the sum of the optimal value functions. Inequality (9) can be interpreted as whether the "benefit" (cost decrease) $-\delta_i$, gained from minimizing the local objective function J_i, is greater than the possible "damage" (cost increase) $\Sigma_{j \in \mathcal{N}_i} \delta_j$ that is done within the objective functions of system i's neighbors. Inequality (10) will be used to ensure that the centralized terminal constraint (6) is satisfied. Note that the calculation of $J_{i,old}$ and $J_{j,test}$ in Step 2) and 3B), respectively, is a pure evaluation of the respective cost function, i.e., no optimization problem has to be solved. Hence each system i has to solve Problem $\mathcal{P}_i$ only once per sampling instant, namely in Step 3A).

The communication requirement for Algorithm 1 is such that at each sampling instant, each system sends data to each of its neighbors four times. Namely, once in Step 1), where each system sends the feasible state trajectory $\hat{\mathbf{x}}_i$ to all neighbors; furthermore in Steps 3B) and 3D), while determining its own optimal input trajectory; and finally in Step 3C) when one of its neighbors computes its optimal input trajectory. If the cost functions are decoupled as in [10], i.e., do not depend on neighboring systems, Algorithm 1 simplifies significantly. Namely, the calculation and exchange of δ_j and e_j in Steps 3B) and 3C) can be omitted in this case, as optimizing the local input trajectory $\mathbf{u}_i$ only affects the local performance criterion. Thus, the communication requirements are also significantly lowered. In Section 3.2, we will show that Algorithm 1 does not only simplify in case of decoupled cost functions, but also for coupled cost functions which exhibit a certain separable structure.

Remark 1 *For clarity of presentation, in Step 3) of Algorithm 1 we require the systems to calculate a new trajectory at each time step in a sequential order. However, the algorithm works in the same way if two systems i and j for which $\mathcal{N}_i \cap \mathcal{N}_j = \emptyset$ do this in parallel. This means that Algorithm 1 is scalable with the number of systems, provided that the number of neighbors of each system is limited.*

3.1.1 Analysis of the DMPC algorithm with general cost

In this section, we show that the proposed distributed MPC algorithm remains feasible for all times if initial feasibility is assumed, and that the resulting closed-loop system is asymptotically stable with respect to the set $\mathcal{X}^0$. To this end, denote the overall closed-loop system resulting from the application of Algorithm 1 by

$$x(k+1) = f(x(k), k^{MPC}(x(k))), \qquad k \in \mathbb{Z}_+, \tag{11}$$

where $k^{MPC} := [(k_1^{MPC})^T, \ldots, (k_N^{MPC})^T]^T$ is the control law obtained in Step 4) of Algorithm 1. Furthermore, define $\mathcal{X}_T \subseteq X$ as the set of all states for which a feasible solution can be found in Step 0) of Algorithm 1. We are now in a position to state the main result of this section:

Theorem 1 *Suppose that an initial solution in Step 0) of Algorithm 1 exists, and that Assumption 1 is satisfied. Then, Problem $\mathcal{P}_i$ in Step 3A) of Algorithm 1 is recursively feasible, for each of the systems $i \in \mathcal{I}$. Furthermore, the overall closed-loop system* (11) *is asymptotically stable with respect to the set $\mathcal{X}^0$ with region of attraction $\mathcal{X}_T$.*

Proof of Theorem 1: We start by proving recursive feasibility of Algorithm 1, which follows the arguments of [10]. First, note that at each time instant k, the feasible state sequences $\hat{\mathbf{x}}_j(k)$ with which $\tilde{\mathbf{x}}_{-i}(k)$ is initialized in (8), are successively replaced by $\mathbf{x}_j^*(k)$. Thus at the end of the loop in Step 3), $\tilde{\mathbf{x}}_{-i}(k)$ is given by

$$\tilde{\mathbf{x}}_{-i}(k) = \left[\mathbf{x}_{i_1}^*(k)^T \ldots \mathbf{x}_{i_{d_i}}^*(k)^T\right]^T =: \mathbf{x}_{-i}^*(k)$$

for all $i \in \mathcal{I}$, i.e., each system knows the latest optimal predicted state sequences of its neighbors. Thus when computing the candidate input sequence $\hat{\mathbf{u}}_i$ in (7) at the next time step $k+1$, each system uses the final states of its neighbors' previously calculated optimal state sequences in its local controller k_i^{loc}.

Now assume that a feasible solution to Problem $\mathcal{P}_i$ exists at time instant $k-1$ for all systems $i \in \mathcal{I}$, such that also the terminal constraint (6) is satisfied. Consider the system with index $i = 1$ at the following time instant k. The candidate input sequence $\hat{\mathbf{u}}_1$ (7) satisfies the input constraint (2d), as it consists of the remaining part of the previously optimal input sequence concatenated with the auxiliary local controller. By the same argumentation, the candidate state sequence $\hat{\mathbf{x}}_1(k)$ satisfies the local state constraint (2e). Furthermore, the terminal state constraint (6) is also satisfied as the terminal region $\mathcal{X}^f$ is invariant under the auxiliary local control laws according to Assumption 1. Finally, consider the coupling constraints (2f). As for $l = k, \ldots, k+T-1$, both $\hat{x}_1(l|k)$ and $\tilde{x}_{-1}(l|k)$ consist of the remaining parts of the previously calculated optimal state sequences, the coupling constraints are fulfilled due to the assumption that they were fulfilled at time instant $k-1$. For $l = k+T$, the coupling constraints are also satisfied according to (4b), as $\hat{x}(k+T|k)$ lies inside the terminal region $\mathcal{X}^f$. Hence the candidate input sequence $\hat{\mathbf{u}}_i$ (7) is a feasible solution to Problem $\mathcal{P}_1$ at time instant k, and consequently also the optimal state trajectory obtained from Problem $\mathcal{P}_1$, $x_{1,test}^*(k)$, satisfies the constraints (2b)–(2f). Furthermore, according to Step 3C), $x_{1,test}^*(k)$ is only assigned to $x_1^*(k)$ if (10) is satisfied, i.e., if the sum of the terminal cost functions when using $x_{1,test}^*(k)$ is less or equal than when the feasible state sequence $\hat{\mathbf{x}}_1(k)$ is used. But this means that with $x_1^*(k)$, also the terminal constraint (6) is satisfied.

Assume now that at time instant k, a feasible solution exists for the systems 1 to i_0, where $1 \leq i_0 \leq n-1$. Then, there also exists a feasible solution at time instant k for the system i_0+1. This can be proven as follows. Input constraint satisfaction (2d) for the candidate input sequence (7) as well as the satisfaction of the local state constraint (2e) for the corresponding candidate state sequence $\hat{\mathbf{x}}_{i_0+1}(k)$ is established as above for system 1. Furthermore, the candidate state trajectory $\hat{\mathbf{x}}_{i_0+1}(k)$ is exactly the trajectory which the previously optimizing neighbors assumed for system i_0+1. Thus the terminal constraint (6) and the coupling constraints (2f) are satisfied. Hence $\hat{\mathbf{u}}_{i_0+1}$ is a feasible solution to Problem $\mathcal{P}_{i_0+1}$ at time instant k, and consequently, as above, with $x^*_{i_0+1}(k)$, the constraints (2b)–(2f) and (6) are satisfied. With this, by induction over i_0, feasibility of Problem $\mathcal{P}_i$ for all systems $i \in \mathcal{I}$ at time instant k can be established, with $x^*(k+T|k)$ satisfying the terminal constraint (6). Feasibility for all times then follows by induction over k and the assumption of initial feasibility.

We now proceed with proving that the overall closed-loop system (11) is asymptotically stable with respect to the set $\mathcal{X}^0$. To this end, consider the functions

$$V_i(k) := J_i(x_i(k), \mathbf{x}^*_{-i}(k), \mathbf{u}^*_i(k))$$

and use $V(x(k)) := \sum_{i=1}^N V_i(k)$ as a Lyapunov function candidate. According to Step 3) of Algorithm 1, it holds that

$$\begin{aligned} V(x(k)) &= \sum_{i \notin \{\mathcal{N}_N \cup \{N\}\}} V_i(k) + \sum_{i \in \mathcal{N}_N} V_i(k) + V_N(k) \\ &\leq \sum_{i \notin \{\mathcal{N}_N \cup \{N\}\}} V_i(k) + \sum_{i \in \mathcal{N}_N} J_i\Big(x_i(k), \begin{bmatrix} \mathbf{x}^*_{i_1}(k) \\ \vdots \\ \mathbf{x}^*_{i_{d_i-1}}(k) \\ \hat{\mathbf{x}}_N(k) \end{bmatrix}, \mathbf{u}^*_i(k)\Big) + J_N(x_N(k), \mathbf{x}^*_{-N}(k), \hat{\mathbf{u}}_N(k)), \end{aligned} \tag{12}$$

where the inequality in (12) holds due to the definition of $\mathbf{x}^*_N(k)$ in Step 3C) of Algorithm 1. Namely, the newly calculated locally optimal state sequence $\mathbf{x}^*_{N,test}(k)$ is only assigned to $\mathbf{x}^*_N(k)$ if (9) holds, which establishes the inequality in (12). On the other hand, if (9) is not satisfied, then $\hat{\mathbf{x}}_N(k)$ is assigned to $\mathbf{x}^*_N(k)$, and thus (12) holds with equality. Using the same argument as in (12) recursively from $i = N$ down to 1, one obtains that

$$V(x(k)) \leq \sum_{i=1}^N J_i(x_i(k), \hat{\mathbf{x}}_{-i}(k), \hat{\mathbf{u}}_i(k)). \tag{13}$$

Thus it holds that for all $x(k) := x \in \mathcal{X}_T$,

$$
\begin{aligned}
V(x(k+1)) - V(x(k)) &\overset{(13)}{\leq} \sum_{i=1}^{N} J_i(x_i(k+1), \hat{\mathbf{x}}_{-i}(k+1), \hat{\mathbf{u}}_i(k+1)) - V(x(k)) \\
&= \sum_{i=1}^{N} \Big(- L_i\big(x_i^*(k|k), x_{-i}^*(k|k), u_i^*(k|k)\big) \\
&\qquad\qquad + \{F_i^+ - F_i + L_i\}\big(x^*(k+T|k), k^{loc}(x^*(k+T|k))\big)\Big) \\
&\overset{(4c)}{\leq} -\sum_{i=1}^{N} L_i\big(x_i^*(k|k), x_{-i}^*(k|k), u_i^*(k|k)\big), \\
&\leq -\alpha_1(|x^*(k|k)|_{\mathcal{X}^0}) = -\alpha_1(|x(k)|_{\mathcal{X}^0}),
\end{aligned} \tag{14}
$$

where for the second inequality we used the fact that $x^*(k+T|k) \in \mathcal{X}^f$, as established above. This means that the set $\mathcal{X}^0$ is attractive for the closed-loop system (11) with region of attraction $\mathcal{X}_T$. Furthermore, it can be shown that there exist functions $\alpha_3, \alpha_4 \in \mathcal{K}_\infty$ such that $V(x) \geq \alpha_3(|x|_{\mathcal{X}^0})$ for all $x \in \mathcal{X}_T$ and $V(x) \leq \alpha_4(|x|_{\mathcal{X}^0})$ for all $x \in \mathcal{X}^f$ [22]. This implies that $|x(k)| \leq \alpha_3^{-1}(\alpha_4(|x_0|_{\mathcal{X}^0}))$ for all $x_0 \in \mathcal{X}^f$, which means that the set $\mathcal{X}^0$ is Lyapunov stable for the closed-loop system (11). Hence the closed-loop system (11) is asymptotically stable with respect to the set $\mathcal{X}^0$ with region of attraction $\mathcal{X}_T$, which concludes the proof of Theorem 1. □

Remark 2 *As pointed out earlier, in Algorithm 1 the centralized terminal constraint* (6) *could not be directly incorporated into the optimization problems $\mathcal{P}_i$, but its satisfaction was ensured in a distributed way via* (10)*. This might result in an unnecessary use of the candidate input sequences $\hat{\boldsymbol{u}}_i$ instead of the newly calculated optimal trajectories $\boldsymbol{u}_{i,test}^*$ in Step 3C) of Algorithm 1. We will show in Section 3.2 how this limitation can be removed if the cost functions exhibit a certain separable structure.*

Remark 3 *Ensuring satisfaction of the terminal constraint* (6) *in the way described above is also conservative due to the reason that via* (10)*, we in fact require the sum of the terminal cost functions to decrease at each sampling instant. However, we can relax this as follows. Suppose that at some time step k, for system i it holds that $e_i + \sum_{j \in \mathcal{N}_i} e_j = -\bar{e}_i$ for some $\bar{e}_i \geq 0$. Then, at the next time step, in order for the terminal constraint to be satisfied it is sufficient that $e_i + \sum_{j \in \mathcal{N}_i} e_j \leq \bar{e}_i$. This procedure can be iterated by updating the "buffer" $\bar{e}_i$ at each time step, thus allowing the sum of the terminal cost functions to possibly increase at later time steps. Another possibility to lower conservatism is that each system i sends $\bar{e}_i$ to its next optimizing neighbor i_{next}. Then, for system i_{next}, in order for the terminal constraint to hold, it is enough that $e_{i_{next}} + \sum_{j \in \mathcal{N}_{i_{next}}} e_j \leq \bar{e}_i$. With these modifications, in our simulation results we observed that ensuring satisfaction of the terminal constraint in the described way did not result in unnecessary conservatism, i.e., in an unnecessary use of the candidate input sequences $\hat{\boldsymbol{u}}_i$ instead of the newly calculated optimal trajectories $\boldsymbol{u}_{i,test}^*$ in Step 3C) of Algorithm 1.*

3.2 DMPC: separable cost functions

In this section, we show how Algorithm 1 can be simplified if the stage and terminal cost functions exhibit a certain separability property. Furthermore, as pointed out in Remark 2, in this case the centralized terminal constraint (6) can be separated and consequently directly included into the optimization problems $\mathcal{P}_i$.

Assume that L_i and F_i are given as

$$\begin{aligned} L_i(x_i, \{x_j\}_{j\in\mathcal{N}_i}, u_i) &= L_{ii}(x_i, u_i) + \sum_{j\in\mathcal{N}_i} L_{ij}(x_i, x_j), \\ F_i(x_i, \{x_j\}_{j\in\mathcal{N}_i}) &= F_{ii}(x_i) + \sum_{j\in\mathcal{N}_i} F_{ij}(x_i, x_j), \end{aligned} \tag{15}$$

i.e., the stage and terminal cost functions of each system are separated into a part L_{ii} (respectively, F_{ii}) consisting of its own state and input, and parts L_{ij} (respectively, F_{ij}) consisting of its own and *one* of its neighbors' states. In order to make use of this separable structure, define the modified optimal control problem $\bar{\mathcal{P}}_i$ for each system i:

Problem $\bar{\mathcal{P}}_i$: At time instant k,

$$\underset{\mathbf{u}_i(k)}{\text{minimize}} \ \bar{J}_i(x_i(k), \tilde{\mathbf{x}}_{-i}(k), \mathbf{u}_i(k))$$

subject to the constraints (2b)–(2f) and the additional terminal constraint

$$x_i(k+T|k) \in \mathcal{X}_i^f(k), \tag{16}$$

where

$$\begin{aligned} \bar{J}_i(x_i(k), \tilde{\mathbf{x}}_{-i}(k), \mathbf{u}_i(k)) := J_i(x_i(k), \tilde{\mathbf{x}}_{-i}(k), \mathbf{u}_i(k)) &+ \sum_{l=0}^{T-1} \sum_{j\in\mathcal{N}_i} L_{ji}(\tilde{x}_j(k+l|k), x_i(k+l|k)) \\ &+ \sum_{j\in\mathcal{N}_i} F_{ji}(\tilde{x}_j(k+T|k), x_i(k+T|k)). \end{aligned} \tag{17}$$

The (time-varying) terminal regions $\mathcal{X}_i^f(k)$ will be specified later on. In Problem $\bar{\mathcal{P}}_i$, each system i minimizes a cost function $\bar{J}_i$ which is comprised of the sum of its own cost function, J_i (defined as in (2a)), and those parts of the cost functions of its neighbors which involve its own state x_i. Hence in the following we assume that system i knows the functions L_{ji} and F_{ji}, for all $j \in \mathcal{N}_i$. The modified DMPC algorithm can now be specified as follows:

Algorithm 2 *Distributed MPC: separable cost functions.*

0) Same as Step 0) of Algorithm 1.

1) Same as Step 1) of Algorithm 1.

2) After receiving the trajectories $\hat{\boldsymbol{x}}_j(k)$ from all of its neighbors $j \in \mathcal{N}_i$, each system i stacks them together into

$$\tilde{\boldsymbol{x}}_{-i}(k) = \left[\hat{\boldsymbol{x}}_{i_1}(k)^T \dots \hat{\boldsymbol{x}}_{i_{d_i}}(k)^T\right]^T =: \hat{\boldsymbol{x}}_{-i}(k).$$

3) For $i = 1$ to N, system i

A) defines the terminal region

$$\mathcal{X}_i^f(k) := \Big\{ y \in \mathcal{X}_i : F_i(y, \tilde{x}_{-i}(k+T|k)) + \sum_{j \in \mathcal{N}_i} F_{ji}(\tilde{x}_j(k+T|k), y) \\ \leq F_i(\hat{x}_i(k+T|k), \tilde{x}_{-i}(k+T|k)) + \sum_{j \in \mathcal{N}_i} F_{ji}(\tilde{x}_j(k+T|k), \hat{x}_i(k+T|k)) \Big\}, \tag{18}$$

B) solves Problem $\bar{\mathcal{P}}_i$ with $\mathcal{X}_i^f(k)$ given by (18)*, and denotes the solution by $\boldsymbol{u}_i^*(k)$ and the corresponding state trajectory by $\boldsymbol{x}_i^*(k)$,*

C) sends $\boldsymbol{x}_i^*(k)$ to its neighbors $j \in \mathcal{N}_i$, who update $\tilde{\boldsymbol{x}}_{-j}(k)$ by replacing $\hat{\boldsymbol{x}}_i(k)$ by $\boldsymbol{x}_i^*(k)$.

4) Each system applies $k_i^{MPC} := u_i^*(k|k)$.

5) Increment k and go to Step 1).

Compared to Algorithm 1, all steps which are needed for checking whether the sum of the cost functions as well as the terminal cost functions decreases (i.e., the exchange of δ_j and e_j, respectively) can be omitted in Algorithm 2. Note that this leads to a significant decrease in the communication requirements, as each system sends data to its neighbors only twice during each sampling instant, namely in Step 2) and 3C), compared to four times in Algorithm 1. The above holds due to the separable structure of the functions L_i and F_i and the special choice of the cost function $\bar{J}_i$ which is minimized within Problem $\bar{\mathcal{P}}_i$, as $\bar{J}_i$ takes into account all terms in any of the systems' cost functions which are affected by the input u_i through the state x_i. From a game theoretic point of view, this can be seen as a cooperative distributed MPC algorithm (see, e.g., [13] for a more detailed discussion on this issue).

3.2.1 Analysis of the DMPC algorithm with separable cost

In this section, we show that under application of the modified DMPC Algorithm 2, recursive feasibility as well as asymptotic stability of the the overall closed-loop system (11) with respect to the set $\mathcal{X}^0$ can be guaranteed, where k^{MPC} now is the control law obtained in Step 4) of Algorithm 2.

Theorem 2 *Suppose that an initial solution in Step 0) of Algorithm 2 exists, and that Assumption 1 is satisfied. Then, Problem $\bar{\mathcal{P}}_i$ in Step 3B) of Algorithm 2 is recursively feasible, for each of the systems $i \in \mathcal{I}$. Furthermore, the overall closed-loop system* (11) *is asymptotically stable with respect to the set $\mathcal{X}^0$ with region of attraction $\mathcal{X}_T$.*

Proof of Theorem 2: Recursive feasibility of Problem $\bar{\mathcal{P}}_i$ can be proven as shown in the proof of Theorem 1, with the exception that the terminal constraint is now directly included into the optimization problem $\bar{\mathcal{P}}_i$. Clearly, the candidate state sequence $\hat{\mathbf{x}}_i(k)$ satisfies the terminal constraint (18), and hence Problem $\bar{\mathcal{P}}_i$ is recursively feasible for all systems i and all time instances k. Furthermore, note that the terminal regions $\mathcal{X}_i^f(k)$ are such that $x^*(k+T|k) \in \mathcal{X}^f$, i.e., the centralized terminal constraint is satisfied. Namely, according to the definition of $\mathcal{X}_i^f(k)$ in (18), the sum of the terminal cost functions when using $x_i^*(k)$ is less or equal than when the feasible state sequence $\hat{\mathbf{x}}_i(k)$ is used, analogously to Algorithm 1, where this was ensured via (10).

In order to prove asymptotic stability of the overall closed-loop system (11) with respect to the set $\mathcal{X}^0$, consider the following. For any system i and any neighbor $k \in \mathcal{N}_i$, define

$$\begin{aligned} L_i^k(x_i, x_{-i}, u_i) &:= L_{ii}(x_i, u_i) + \sum_{j \in \mathcal{N}_i, j \neq k} L_{ij}(x_i, x_j) = L_i(x_i, x_{-i}, u_i) - L_{ik}(x_i, x_k), \\ F_i^k(x_i, x_{-i}) &:= F_{ii}(x_i) + \sum_{j \in \mathcal{N}_i, j \neq k} F_{ij}(x_i, x_j) = F_i(x_i, x_{-i}) - F_{ik}(x_i, x_k), \\ J_i^k(x_i(k), \tilde{\mathbf{x}}_{-i}(k), \mathbf{u}_i(k)) &:= \sum_{l=0}^{T-1} L_i^k\big(x_i(k+l|k), \tilde{x}_{-i}(k+l|k), u_i(k+l|k)\big) \\ &\quad + F_i^k\big(x_i(k+T|k), \tilde{x}_{-i}(k+T|k)\big). \end{aligned} \tag{19}$$

Now consider again the functions

$$V_i(k) := J_i(x_i(k), \mathbf{x}_{-i}^*(k), \mathbf{u}_i^*(k))$$

and use $V(x(k)) := \sum_{i=1}^N V_i(k)$ as a Lyapunov function candidate. We obtain

$$\begin{aligned} V(x(k)) = &\sum_{i \notin \{\mathcal{N}_N \cup \{N\}\}} V_i(k) + \sum_{i \in \mathcal{N}_N} V_i(k) + V_N(k) \\ \overset{(19)}{=} &\sum_{i \notin \{\mathcal{N}_N \cup \{N\}\}} V_i(k) + \sum_{i \in \mathcal{N}_N} J_i^N(x_i(k), \mathbf{x}_{-i}^*(k), \mathbf{u}_i^*(k)) \\ &+ \sum_{l=0}^{T-1} \sum_{i \in \mathcal{N}_N} L_{iN}(x_i^*(k+l|k), x_N^*(k+l|k)) \\ &+ \sum_{i \in \mathcal{N}_i} F_{iN}(x_i^*(k+T|k), x_N^*(k+T|k)) + J_N(x_N(k), \mathbf{x}_{-N}^*(k), \mathbf{u}_N^*(k)) \\ \overset{(17)}{=} &\sum_{i \notin \{\mathcal{N}_N \cup \{N\}\}} V_i(k) + \sum_{i \in \mathcal{N}_N} J_i^N(x_i(k), \mathbf{x}_{-i}^*(k), \mathbf{u}_i^*(k)) + \bar{J}_N(x_N(k), \mathbf{x}_{-N}^*(k), \mathbf{u}_N^*(k)). \end{aligned} \tag{20}$$

But as in Step 3A) of Algorithm 2, system N solves the optimization problem $\bar{\mathcal{P}}_N$ with objective function $\bar{J}_N$, due to optimality of $\mathbf{u}_N^*$ we obtain

$$\begin{aligned} \bar{J}_N(x_N(k), \mathbf{x}_{-N}^*(k), \mathbf{u}_N^*(k)) \leq \bar{J}_N(x_N(k), \mathbf{x}_{-N}^*(k), \hat{\mathbf{u}}_N(k)) \\ \overset{(17)}{=} J_N(x_N(k), \mathbf{x}_{-N}^*(k), \hat{\mathbf{u}}_N(k)) + \sum_{l=0}^{T-1} \sum_{i \in \mathcal{N}_N} L_{iN}(x_i^*(k+l|k), \hat{x}_N(k+l|k)) \\ + \sum_{i \in \mathcal{N}_i} F_{iN}(x_i^*(k+T|k), \hat{x}_N(k+T|k)). \end{aligned}$$

Plugging this into (20) and using again (19) yields

$$V(x(k)) \le \sum_{i \notin \{\mathcal{N}_N \cup \{N\}\}} V_i(k) + \sum_{i \in \mathcal{N}_N} J_i(x_i(k), \begin{bmatrix} \mathbf{x}^*_{i_1}(k) \\ \vdots \\ \mathbf{x}^*_{i_{d_i-1}}(k) \\ \hat{\mathbf{x}}_N(k) \end{bmatrix}, \mathbf{u}^*_i(k)) + J_N(x_N(k), \mathbf{x}^*_{-N}(k), \hat{\mathbf{u}}_N(k)). \tag{21}$$

Using the same argument as in (21) recursively from $i = N$ down to 1, one again obtains that (13) holds, and the rest of the proof then follows along the lines of the proof of Theorem 1. □

Remark 4 *Similar to the situation described in Remark 3, one can incorporate some additional buffer $\bar{e}_i(k)$ into the definition of the terminal regions $\mathcal{X}_i^f(k)$ to reduce possible conservatism. Namely, if the terminal regions $\mathcal{X}_i^f(k)$ are defined as in (18) with $\bar{e}_i(k)$ added on the right hand side of the inequality, the proof of Theorem 2 still works in the same way provided that the buffer $\bar{e}_i(k)$ is updated according to one of the methods described in Remark 3.*

Both Algorithm 1 and 2 are rather general in nature and can be used for the solution of a number of typical distributed control problems. In the following two sections, we will specifically show how they can be applied for (i) the task of stabilizing a prespecified setpoint (Section 4) and (ii) consensus and synchronization problems (Section 5). In particular, we show how the terminal region $\mathcal{X}^f$ and the auxiliary local controllers k_i^{loc} can be computed such that Assumption 1 is satisfied.

4 Application of DMPC algorithms to setpoint stabilization

In this section, we show how the proposed DMPC algorithms can be used for the stabilization of an a priori known setpoint, while optimizing some common goal for neighboring trajectories. We first show that in such a setting, fixed decoupled terminal regions can be computed, which can directly be incorporated into the optimization problems $\mathcal{P}_i$ and $\bar{\mathcal{P}}_i$, respectively. With this, Algorithm 1 and 2 are simplified as the computation of the time-varying terminal regions $\mathcal{X}_i^f(k)$ in Step 3A) of Algorithm 2, as well as the computation and exchange of the terminal cost function differences e_j in Algorithm 1 become obsolete. After that, we will illustrate the results with a specific example and compare it to other DMPC schemes in the literature.

4.1 Computation of decoupled terminal regions

Without loss of generality, suppose that the origin is an equilibrium point for each of the systems (1), and that the goal is to asymptotically stabilize this setpoint for all of the systems. To this end, suppose that the stage cost functions L_i for the systems $i \in \mathcal{I}$ are given by

$$L_i(x_i, \{x_j\}_{j \in \mathcal{N}_i}, u_i) := x_i^T Q_i x_i + u_i^T R_i u_i + \sum_{j \in \mathcal{N}_i} (C_i x_i - C_j x_j)^T Q_{ij} (C_i x_i - C_j x_j), \tag{22}$$

where $Q_i, R_i > 0$, $Q_{ij} \geq 0$, and C_i, C_j are matrices of appropriate dimensions. The coupling terms in the sum can be interpreted in the following way: Some output $y_i := C_i x_i$ of system i should enter the origin along the same trajectory as the output $y_j := C_j x_j$ of some neighbor j, and the matrix Q_{ij} penalizes the deviation. The type of cost function (22) can be used to consider several applications, including e.g. multi-vehicle formation stabilization and others [8, 9].

For the described setting, a centralized terminal region and auxiliary local controllers satisfying Assumption 1 can be calulated using standard techniques (see, e.g., [6]), and Algorithms 1 and 2 can readily be applied. However, in the considered case of setpoint stabilization, also decoupled terminal regions $\mathcal{X}_i^f$ and auxiliary local controllers $k_i^{loc}(x_i)$, only depending on x_i, can be calculated such that Assumption 1 is satisfied with $\mathcal{X}^f = \mathcal{X}_1^f \times \cdots \times \mathcal{X}_n^f$. While this might lead to some conservatism in the size of the terminal regions (due to the upper-bounding estimate in (23) below), it allows us to simplify Algorithm 1 and 2, respectively, as explained above. Now consider the coupling terms in the stage cost (22). It holds that

$$\begin{aligned}\sum_{j\in\mathcal{N}_i} (y_i - y_j)^T Q_{ij}(y_i - y_j) &= \sum_{j\in\mathcal{N}_i} y_i^T Q_{ij} y_i - y_i^T Q_{ij} y_j - y_j^T Q_{ij} y_i + y_j^T Q_{ij} y_j \\ &\leq \sum_{j\in\mathcal{N}_i} y_i^T Q_{ij} y_i + \lambda_{max}(Q_{ij})(y_i^T y_i + y_j^T y_j) + y_j^T Q_{ij} y_j \\ &= \sum_{j\in\mathcal{N}_i} y_i^T (Q_{ij} + \lambda_{max}(Q_{ij}) I) y_i + y_j^T (Q_{ij} + \lambda_{max}(Q_{ij}) I) y_j. \qquad (23)\end{aligned}$$

We can now decouple the calculation of the auxiliary local controllers and the terminal regions by letting system i take care of the terms in (23) involving y_i, and system j taking care of the terms involving y_j. Thus, if the auxiliary local controllers and the terminal regions are calculated such that

$$\begin{aligned}&F_i\big(f_i(x_i, k_i^{loc}(x_i))\big) - F_i(x_i) \\ &\leq -\Big(x_i^T Q_i x_i + k_i^{loc}(x_i)^T R_i k_i^{loc}(x_i) + y_i^T \big(\sum_{j\in\mathcal{N}_i} Q_{ij} + Q_{ji} + (\lambda_{max}(Q_{ij}) + \lambda_{max}(Q_{ji}))I\big) y_i\Big) \\ &=: -\tilde{L}_i(x_i, k_i^{loc}(x_i))\end{aligned}$$

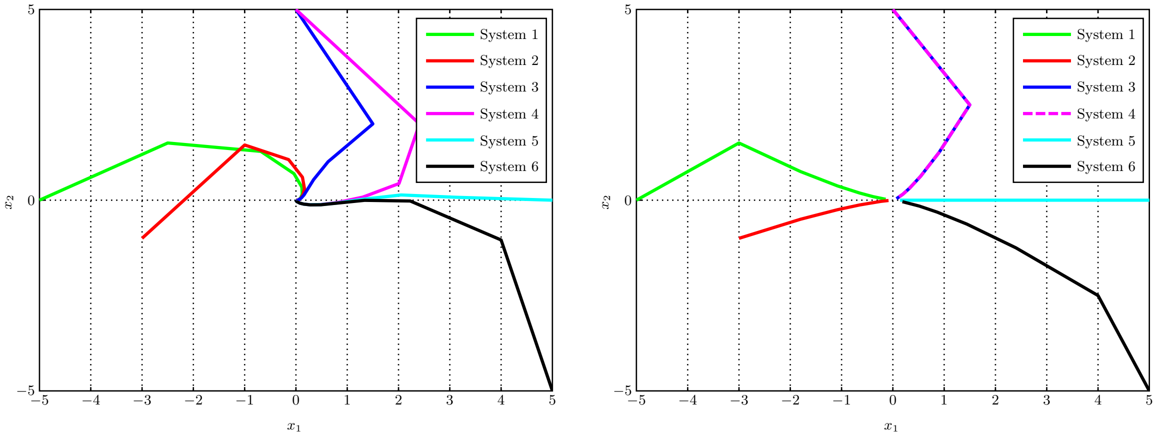
for all $x_i \in \mathcal{X}_i^f$, then Condition (4c) in Assumption 1 is satisfied. But as $\tilde{L}_i$ only depends on x_i, i.e., is decoupled from the neighboring systems, this can be done using standard techniques, e.g., by considering the linearization around the origin [6].

4.2 Example

As specific example, we consider the task to stabilize the origin of six two-dimensional nonlinear systems, with a line graph as underlying interconnection topology. For all $i \in \mathcal{I}$, $\mathcal{X}_i = \{x \in \mathbb{R}^2 | \|x\|_\infty \leq 5\}$, $\mathcal{U}_i = \{u \in \mathbb{R}^2 | \|u\|_\infty \leq 2\}$, and the system dynamics are given by

$$x_i(k+1) = \begin{bmatrix} 0.9 & 0.2 \\ -0.2 & 0.8 \end{bmatrix} x_i(k) + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} u_i(k) + 0.1 \begin{bmatrix} x_{i,2}(k)^2 \\ x_{i,1}(k)^2 \end{bmatrix}. \qquad (24)$$

The stage cost is given as in (22) with $Q_i = I_2$, $R_i = 5I_2$ and $C_i = I_2$ for all i; furthermore, $Q_{34} = Q_{43} = 0_2$ and all other Q_{ij} were chosen as $Q_{ij} = 3I_2$. One coupling constraint is assumed to



(a) State trajectories using DMPC.

(b) Initially planned trajectories.

Figure 1: Setpoint stabilization via DMPC in Section 4.2.

be present coupling Systems 3 and 4 such that $\|x_3 - x_4\|_2 \leq 4$. The terminal controllers and terminal regions were calculated as described above such that Assumption 1 is satisfied. Figure 1 shows the simulation results when applying Algorithm 2. It can be seen that the origin is asymptotically stabilized, and that Systems 1 to 3 as well as 4 to 6 enter the origin on close by trajectories, as desired. Furthermore, the coupling constraint is satisfied for all times. Other DMPC schemes which can also take into account coupled cost functions like [8] have a "consistency constraint" added to the optimization problem, ensuring that the newly calculated optimal trajectory does not deviate too far from the previously calculated optimal trajectory, which is needed to ensure asymptotic stability. Thus the solution heavily depends on the initially found trajectories and the actual trajectories cannot deviate too much from these. As can be seen from the comparison of the actual state trajectories when applying Algorithm 2 (Figure 1(a)) with the initially planned trajectories (Figure 1(b)), this is not the case when Algorithm 1 is applied, i.e., the actual state trajectories can deviate quite far from the initially assumed trajectories. Furthermore, hard coupling constraints can be taken into account, which is not the case in [8, 9, 11] and constitutes an advantage of the present approach.

5 Application of DMPC algorithms to consensus and synchronization

In this section, we show how the proposed DMPC algorithms can be used to reach state consensus between the systems, i.e., our goal is to synchonize the trajectories of the systems. This can be translated in that the overall closed-loop system (11) is asymptotically stable with respect to the set $\mathcal{X}^0 := \{x \in \mathcal{X} : x_1 = x_2 = \cdots = x_N\}$. To this end, in the following we will consider a network of N homogeneous agents, i.e., systems with identical dynamics. We first show how the terminal region $\mathcal{X}^f$ and the auxiliary local controllers k_i^{loc} can be calculated such that Assumption 1 is satisfied in case of linear system dynamics and quadratic cost functions, and then extend the results to the nonlinear case. In both cases, we will illustrate our results with examples of synchronizing oscillators.

In the following, we will make use of the incidence matrix $E(\mathcal{G}) \in \mathbb{R}^{N \times M}$ of the graph $\mathcal{G}$ specifying the interconnection topology of the systems. In order to define $E(\mathcal{G})$, we assign an arbitrary orientation to each edge, i.e. such that each edge has a head (terminal node) and a tail (initial node). The columns of $E(\mathcal{G})$ are now indexed by the edge set and the rows by the set of nodes, and the i-th row entry takes the value $+1$ if node i is the initial node of the corresponding edge, -1 if it is the terminal node, and 0 otherwise [23].

5.1 Linear systems

In this section, we consider the special case of a network of N systems with identical linear dynamics, i.e., (1) reduces to

$$s_i : \qquad x_i(k+1) = Ax_i(k) + Bu_i(k), \qquad k \in \mathbb{Z}_+. \tag{25}$$

Define $\xi_\ell := x_i - x_j$ as the state difference of the systems s_i and s_j corresponding to the ℓ-th column of $E(\mathcal{G})$ indexed by the edge (s_i, s_j). Let $\xi := [\xi_1^T, \dots, \xi_M^T]^T$, and note that $\xi = (E(\mathcal{G})^T \otimes I_n)x$ [23]. Now consider the local auxiliary controller candidates

$$k_i^{loc} = \sum_{\ell \in \mathcal{N}_i^t} K_\ell \xi_\ell - \sum_{\ell \in \mathcal{N}_i^h} K_\ell \xi_\ell, \tag{26}$$

where $\mathcal{N}_i^t$ denotes the set of edges with node i being the tail, and $\mathcal{N}_i^h$ the set of edges with node i being the head. Note that k_i^{loc} defined by (26) is of the form $k_i^{loc} = k_i^{loc}(x_i, \{x_j\}_{j \in \mathcal{N}_i})$, as required in Assumption 1. From (26), one obtains that $k^{loc} := [(k_1^{loc})^T, \dots, (k_N^{loc})^T]^T$ can be expressed as

$$k^{loc}(x) = (E(\mathcal{G}) \otimes I_m)K\xi = (E(\mathcal{G}) \otimes I_m)K(E(\mathcal{G})^T \otimes I_n)x, \tag{27}$$

where $K = diag(K_\ell)$. With this, the overall system in closed-loop with the local auxiliary control law can be written as

$$\begin{aligned} x(k+1) &= (I_N \otimes A)x(k) + (I_N \otimes B)k^{loc}(k) \\ &= (I_N \otimes A)x(k) + (I_N \otimes B)(E(\mathcal{G}) \otimes I_m)K(E(\mathcal{G})^T \otimes I_n)x(k) \\ &= \Big((I_N \otimes A) + (I_N \otimes B)(E(\mathcal{G}) \otimes I_m)K(E(\mathcal{G})^T \otimes I_n)\Big)x(k). \end{aligned} \tag{28}$$

Furthermore, by noting that

$$(E(\mathcal{G})^T \otimes I_n)(I_N \otimes A) = E(\mathcal{G})^T I_N \otimes I_n A = I_M E(\mathcal{G})^T \otimes A I_n = (I_M \otimes A)(E(\mathcal{G})^T \otimes I_n), \tag{29}$$

we obtain from (28) that

$$\begin{aligned} \xi(k+1) &= (E(\mathcal{G})^T \otimes I_n)x(k+1) \\ &= (E(\mathcal{G})^T \otimes I_n)\Big((I_N \otimes A) + (I_N \otimes B)(E(\mathcal{G}) \otimes I_m)K(E(\mathcal{G})^T \otimes I_n)\Big)x(k) \\ &= \Big((I_M \otimes A) + (E(\mathcal{G})^T \otimes I_n)(I_N \otimes B)(E(\mathcal{G}) \otimes I_m)K\Big)\xi(k). \end{aligned} \tag{30}$$

Now consider the stage and terminal cost functions given by

$$L_i(x_i, \{x_j\}_{j\in\mathcal{N}_i}, u_i) := u_i^T R_i u_i + \sum_{\ell\in\mathcal{N}_i^t\cup\mathcal{N}_i^h} \xi_\ell^T Q_\ell \xi_\ell, \tag{31}$$

$$F_i(x_i, \{x_j\}_{j\in\mathcal{N}_i}) := \sum_{\ell\in\mathcal{N}_i^t\cup\mathcal{N}_i^h} \xi_\ell^T P_\ell \xi_\ell, \tag{32}$$

where R_i, Q_ℓ, P_ℓ are positive definite matrices of appropriate dimension. Note that L_i and F_i given by (31) and (32), respectively, are of the form (15), and positive definite with respect to the set $\mathcal{X}^0$. Our goal is to compute the local auxiliary controller gains K_ℓ as well as the terminal weighting matrices P_ℓ such that condition (4c) of Assumption 1 is satisfied. With $P := diag(P_\ell)$, $Q := diag(Q_\ell)$, $R := diag(R_i)$, it holds that

$$\sum_{i=1}^N F_i(x_i, \{x_j\}_{j\in\mathcal{N}_i}) = \sum_{i=1}^N \sum_{\ell\in\mathcal{N}_i^t\cup\mathcal{N}_i^h} \xi_\ell^T P_\ell \xi_\ell = 2\xi^T P\xi,$$

$$\sum_{i=1}^N L_i(x_i, \{x_j\}_{j\in\mathcal{N}_i}, k_i^{loc}) = \sum_{i=1}^N (k_i^{loc})^T R_i k_i^{loc} + \sum_{i=1}^N \sum_{\ell\in\mathcal{N}_i^t\cup\mathcal{N}_i^h} \xi_\ell^T Q_\ell \xi_\ell = (k^{loc})^T R k^{loc} + 2\xi^T Q\xi. \tag{33}$$

With this, we can define the terminal region $\mathcal{X}^f$ according to (5) as

$$\mathcal{X}^f := \{x : 2\xi^T P\xi \leq \alpha\} \tag{34}$$

for some $\alpha > 0$. Furthermore, (4c) results in

$$\begin{aligned}
&\sum_{i=1}^N \big\{F_i^+ - F_i + L_i\big\}(x, k^{loc}(x)) \\
&= \xi^T\Big(2\big[\dots\big]^T P\big[(I_M\otimes A) + (E(\mathcal{G})^T\otimes I_n)(I_N\otimes B)(E(\mathcal{G})\otimes I_m)K\big] \\
&\quad - 2P + K^T(E(\mathcal{G})\otimes I_m)^T R(E(\mathcal{G})\otimes I_m)K + 2Q\Big)\xi \leq 0,
\end{aligned} \tag{35}$$

for all $x \in \mathcal{X}^f$. Denoting $\tilde{A} := (I_M\otimes A)$, $\tilde{K} := (E(\mathcal{G})\otimes I_m)K$ and $\tilde{B} := (E(\mathcal{G})^T\otimes I_n)(I_N\otimes B)$, this results in

$$\xi^T\Big(2(\tilde{A}+\tilde{B}\tilde{K})^T P(\tilde{A}+\tilde{B}\tilde{K}) - 2P + \tilde{K}^T R\tilde{K} + 2Q\Big)\xi \leq 0 \tag{36}$$

for all ξ. Inequality (36) can be rewritten as the following equivalent LMI with $X := P^{-1} > 0$ and $Y := \tilde{K}X$ by using standard manipulations such as the Schur complement and left and right multiplying with P^{-1} (see, e.g., [24]):

$$\begin{bmatrix} 2X & \sqrt{2}(X\tilde{A}^T + Y^T\tilde{B}^T) & \sqrt{2}XQ^{1/2} & Y^T R^{1/2} \\ \sqrt{2}(\tilde{A}X + \tilde{B}Y) & X & 0 & 0 \\ \sqrt{2}Q^{1/2}X & 0 & I & 0 \\ R^{1/2}Y & 0 & 0 & I \end{bmatrix} \geq 0. \tag{37}$$

Due to the block-diagonal structure of P, also X is block-diagonal and thus Y has the same structure as $\tilde{K}$. Hence $\tilde{K}$ can be recovered from the solution of (37) as $\tilde{K} = YX^{-1} = YP$, from which K can be obtained by recalling that $\tilde{K}$ was defined as $\tilde{K} := (E(\mathcal{G}) \otimes I_m)K$ and K is block-diagonal, which means that the ℓ-th diagonal block of K can be recovered from the ℓ-th block-column of $\tilde{K}$ due to the specific structure of the incidence matrix $E(\mathcal{G})$.

In case that the graph $\mathcal{G}$ contains cycles, some of the state differences ξ_ℓ are linearly dependent, namely those corresponding to a circle subgraph. To be more precise, the linear dependencies of the state differences ξ_ℓ are given by $e_{null}^T \xi = 0$, where e_{null} is a vector in the nullspace of $E(\mathcal{G})$ [23, 25]. Let us quickly illustrate this property with a simple example. Consider the cycle graph C_3 with three nodes and edges and incidence matrix given by

$$E(C_3) = \begin{bmatrix} 1 & 1 & 0 \\ -1 & 0 & 1 \\ 0 & -1 & -1 \end{bmatrix}, \tag{38}$$

i.e., $\xi_1 = x_1 - x_2$, $\xi_2 = x_1 - x_3$, and $\xi_3 = x_2 - x_3$. Clearly, it holds that $\xi_2 = \xi_1 + \xi_3$, i.e., $e_{null}^T \xi = [1\ \ -1\ 1]\xi = 0$, where $e_{null} = [1\ \ -1\ 1]^T$ spans the nullspace of $E(C_3)$.

Considering the above, it follows that in order for condition (4c) of Assumption 1 to be satisfied, (36) only has to hold for those ξ such that $E_{null}^T \xi = 0$, where E_{null} is a basis for the nullspace of $E(\mathcal{G})$. According to Finsler's Lemma [26] (see also [27]), this is equivalent to the existence of a constant $\rho > 0$ such that

$$2(\tilde{A} + \tilde{B}\tilde{K})^T P(\tilde{A} + \tilde{B}\tilde{K}) - 2P + \tilde{K}^T R\tilde{K} + 2Q - \rho E_{null} E_{null}^T \leq 0. \tag{39}$$

However, (39) cannot be transformed into an LMI analogously to (37), as the additional term $\rho E_{null} E_{null}^T$ in (39) would result in an additional term $\rho X E_{null} E_{null}^T X$ in the first diagonal element of (37), which is quadratic in X and hence destroys the LMI structure. Nevertheless, we can tighten (39) in the following way in order still to obtain an LMI:

$$2(\tilde{A} + \tilde{B}\tilde{K})^T P(\tilde{A} + \tilde{B}\tilde{K}) - 2P + \tilde{K}^T R\tilde{K} + 2Q - \rho(P E_{null} E_{null}^T + E_{null} E_{null}^T P) \leq 0 \tag{40}$$

for some ρ. Note that solvability of (40) implies solvability of (39). Namely, (40) implies that (36) holds for all ξ such that $E_{null}^T \xi = 0$, which, as explained above, is equivalent to (39) according to Finsler's Lemma. In fact, the two inequalities (39) and (40) would be equivalent if in (40) we did not multiply the terms involving $E_{null} E_{null}^T$ with P but rather allowed for an arbitrary positive definite matrix W instead. Now (40) can be transformed into an equivalent LMI (for fixed ρ) analogously to above, and we obtain

$$\begin{bmatrix} 2X + \rho(E_{null} E_{null}^T X + X E_{null} E_{null}^T) & \sqrt{2}(X\tilde{A}^T + Y^T \tilde{B}^T) & \sqrt{2}XQ^{1/2} & Y^T R^{1/2} \\ \sqrt{2}(\tilde{A}X + \tilde{B}Y) & X & 0 & 0 \\ \sqrt{2}Q^{1/2}X & 0 & I & 0 \\ R^{1/2}Y & 0 & 0 & I \end{bmatrix} \geq 0. \tag{41}$$

Our simulation results show that although (40) (and hence also the equivalent LMI (41)) is more restrictive than (39), it still offers a considerable relaxation compared to the LMI (37), where the linear dependency of the state differences ξ_ℓ due to the cycles in the graph was not considered. Furthermore, note that in the case when the graph contains no cycles, it follows that E_{null} is

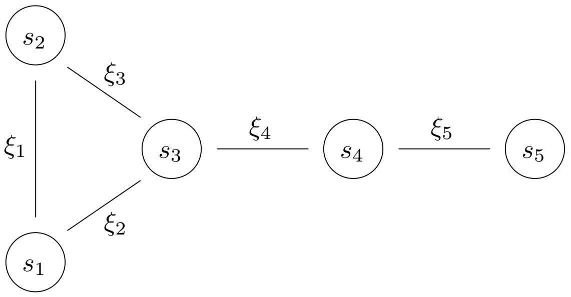


Figure 2: Interconnection topology of the linear oscillators s_i in Section 5.1.1.

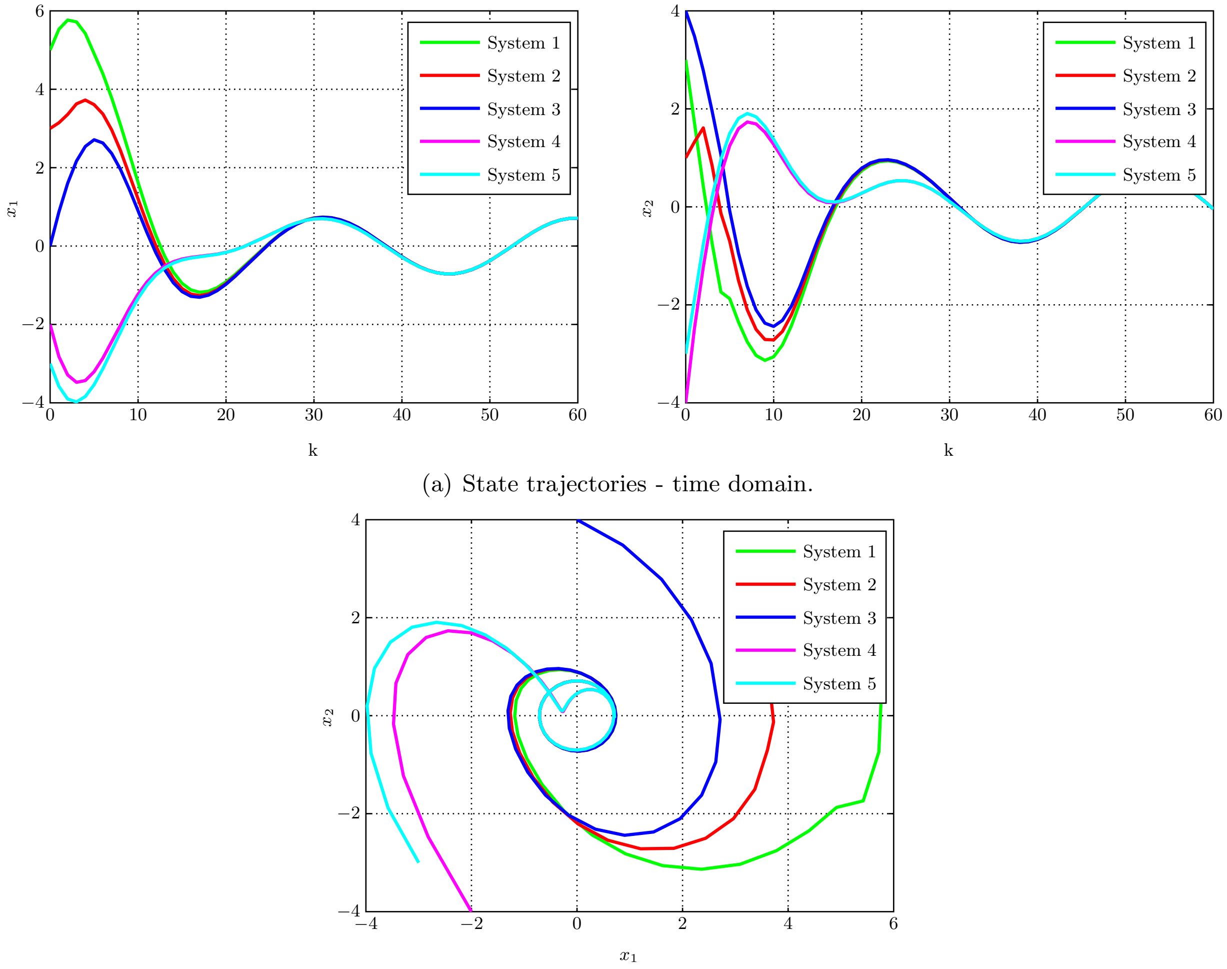
empty, and hence from (41) we recover the LMI (37). Again, as explained above, the local auxiliary controller gains K_ℓ and the terminal weighting matrices P_ℓ can be recovered from the solution X and Y of the LMI (41). We thus arrive at the following result:

Theorem 3 *If the local auxiliary controller gains K_ℓ and the terminal weighting matrices P_ℓ are such that they are obtained from the solution of the LMI* (41)*, then condition* (4c) *of Assumption 1 is satisfied.*

In view of Theorem 3, one can conclude that if no state and input constraints are present, Assumption 1 is satisfied in the whole state space, which thus can be used as terminal region $\mathcal{X}^f$, i.e., we can let $\alpha \to \infty$ in (34). If input constraints are present, α in (34) has to be chosen sufficiently small such that (4a) is satisfied inside the terminal region. Also, for reasonable coupling constraints, it can be assumed that (4b) holds if the terminal region is chosen sufficiently small. Finally, if also local state constraints (2e) should be considered, the situation is not as straightforward. Namely, for any $\alpha > 0$, it is not a priori clear that $\mathcal{X}^f \subseteq \mathcal{X}$, as $\mathcal{X}^f$ defined by (34) is only given as a weighted sum of relative distances between neighboring systems. Hence one has to ensure that $\mathcal{X}^f \cap \mathcal{X}$, or more general, $\mathcal{X}^f \cap \bar{\mathcal{X}}$ for some $\bar{\mathcal{X}} := \{\bar{\mathcal{X}}_1 \times \cdots \times \bar{\mathcal{X}}_N\} \subseteq \mathcal{X}$ is invariant under the local auxiliary control k^{loc}. Then, the local state constraints (2e) are satisfied if the additional terminal constraint $x_i(k+T|k) \in \bar{\mathcal{X}}_i$ is added to the optimization problem $\mathcal{P}_i$ (or $\bar{\mathcal{P}}_i$, respectively). In general, determining such a set $\bar{\mathcal{X}}$ might be a difficult task, and the development of a general design procedure is out of the scope of this paper. In certain applications, on the other hand, like in the example considered in Section 5.2.1, this can be achieved at least numerically.

5.1.1 Example - synchronization of linear oscillators

Consider as an illustrative example the problem of synchronizing five identical linear oscillators. The dynamics of the oscillators s_i are given by (25) with $A = \begin{bmatrix} 0.9762 & 0.2169 \\ -0.2169 & 0.9762 \end{bmatrix}$ and $B = [0\ 1]^T$, and the interconnection topology of the oscillators is specified by the graph $\mathcal{G}$, which is depicted in Figure 2 together with the state differences ξ_ℓ corresponding to the columns of $E(\mathcal{G})$. The input constraints are such that $|u_i| \leq 1$, and the weighting matrices for the stage cost (31) are chosen as $Q_1 = Q_2 = Q_3 = Q_5 = 3I_2$, $Q_4 = 0.1I_2$, and $R_1 = R_2 = R_2 = R_4 = R_5 = 1$. The controller



(a) State trajectories - time domain.

(b) State trajectories - phase plane.

Figure 3: Synchronization of linear oscillators via DMPC in the example of Section 5.1.1.

gains K_ℓ and matrices P_ℓ are calculated according to the above procedure such that Assumption 1 is satisfied. Figure 3 shows simulation results obtained with prediction horizon $T = 15$. One can see that the systems synchronize, and that ξ_4 converges to 0 much slower than the other ξ_ℓ, which is in accordance with the choice the weighting matrices Q_ℓ.

5.2 Nonlinear systems

In this section, we show how the procedure developed in Section 5.1 for linear systems can be extended to nonlinear systems, i.e., how the terminal region and the terminal controllers can be calculated such that Assumption 1 holds. Our approach will be to linearize the systems and then let the terminal region be sufficiently small such that (4c) still holds for the nonlinear system. However, in contrast to [6], where such a method was developed for the task to centrally stabilize an a priori known setpoint, in our setup the point around which the linearized system dynamics are considered is not determined a priori.

Consider a network of N systems with identical nonlinear dynamics, i.e., (1) reduces to

$$s_i: \qquad x_i(k+1) = f(x_i(k), u_i(k)), \qquad k \in \mathbb{Z}_+. \tag{42}$$

Let the stage and terminal cost functions be given by (31)–(32), and the terminal region $\mathcal{X}^f$ by (34). Our goal is that the overall system state x asymptotically converges to the set $\mathcal{X}^0 \cap \bar{\mathcal{X}}$, where $\bar{\mathcal{X}} \subseteq \mathcal{X}$ is a set wherein the synchronized trajectories are restricted to lie. The set $\bar{\mathcal{X}}$ has to be computed such that $\mathcal{X}^f \cap \bar{\mathcal{X}}$ is invariant under the local auxiliary control k^{loc}, as discussed in Section 5.1. For technical reasons which will become apparent later on, we assume $\bar{\mathcal{X}}$ to be compact. Note that in many cases, this is not a major restriction, like in the example considered in Section 5.2.1, where the goal is to synchronize nonlinear oscillators to a bounded limit cycle.

In order to extend the approach developed in Section 5.1 to nonlinear systems, consider the following. For given $x := [x_1^T, \dots, x_N^T]^T$, the arithmetic mean $\bar{x}$ of all agents is given by

$$\bar{x} = \frac{x_1 + \dots + x_N}{N}. \tag{43}$$

Furthermore, for any agent i it holds that

$$\begin{aligned} x_i - \bar{x} &= \frac{N x_i - (x_1 + \dots + x_N)}{N} \\ &= \frac{(x_i - x_1) + \dots + (x_i - x_{i-1}) + (x_i - x_{i+1}) + \dots + (x_i - x_N)}{N}. \end{aligned} \tag{44}$$

As we assume the graph $\mathcal{G}$ to be connected, the difference between the states of two arbitrary agents i and k can be expressed as a linear combination of the state differences ξ_ℓ, $1 \le \ell \le M$, i.e. $x_i - x_k = c_{ik}^T \xi$ for some coefficient vector c_{ik}. Using this, from (44) we obtain that for all i

$$x_i - \bar{x} = \frac{1}{N} \sum_{k \neq i} c_{ik} \xi =: \frac{1}{N} \bar{c}_i \xi. \tag{45}$$

Linearization of the systems (42) around $(\bar{x}, 0)$ yields

$$\begin{aligned} f(x_i, u_i) &= f(\bar{x}, 0) + \frac{\partial f}{\partial x_i}|_{(\bar{x},0)}(x_i - \bar{x}) + \frac{\partial f}{\partial u_i}|_{(\bar{x},0)} u_i + \Phi_{\bar{x}}(x_i - \bar{x}, u_i) \\ &=: f(\bar{x}, 0) + A_{\bar{x}}(x_i - \bar{x}) + B_{\bar{x}} u_i + \Phi_{\bar{x}}(x_i - \bar{x}, u_i), \end{aligned} \tag{46}$$

where Φ constitutes the higher order terms in $(x_i - \bar{x}, u_i)$. The overall linearized system is then given by

$$f_{tot}(x, u) = 1_N \otimes f(\bar{x}, 0) + (I_N \otimes A_{\bar{x}})(x - (1_N \otimes \bar{x})) + (I_N \otimes B_{\bar{x}})u + \bar{\Phi}_{\bar{x}}(x, u), \tag{47}$$

where $\bar{\Phi}_{\bar{x}}(x, u) := [\Phi_{\bar{x}}(x_1 - \bar{x}, u_1)^T, \dots, \Phi_{\bar{x}}(x_N - \bar{x}, u_N)^T]^T$.

Considering (45), it follows that under application of the local auxiliary controller candidates (26), it holds that

$$\bar{\Phi}_{\bar{x}}(x, k^{loc}) = O(|\xi|^2). \tag{48}$$

Hence when applying the local auxiliary controller candidates (26), from (47) and using (29) we obtain

$$\begin{aligned}\xi(k+1) &= (E(\mathcal{G})^T \otimes I_n)x(k+1) = (E(\mathcal{G})^T \otimes I_n)f_{tot}(x(k), k^{loc}(x(k)))\\ &= (\tilde{A}_{\bar{x}} + \tilde{B}_{\bar{x}}\tilde{K})\xi(k) + (E(\mathcal{G})^T \otimes I_n)\bar{\Phi}_{\bar{x}}(x(k), k^{loc}(x(k)))\\ &=: \tilde{A}_{K,\bar{x}}\xi(k) + \Psi_{\bar{x}}(\xi(k)),\end{aligned} \qquad (49)$$

where $\tilde{A}_{\bar{x}} := (I_M \otimes A_{\bar{x}})$, $\tilde{B}_{\bar{x}} := (E(\mathcal{G})^T \otimes I_n)(I_N \otimes B_{\bar{x}})$, and $\Psi_{\bar{x}}(\xi)$ is of order $O(|\xi|^2)$ due to (48). Hence there exist constants $\kappa, \alpha > 0$ such that

$$(\tilde{A}_{K,\bar{x}}\xi + \Psi_{\bar{x}}(\xi))^T P(\tilde{A}_{K,\bar{x}}\xi + \Psi_{\bar{x}}(\xi)) \leq \xi^T \tilde{A}_{K,\bar{x}}^T P \tilde{A}_{K,\bar{x}}\xi + \kappa\xi^T P\xi, \qquad \forall \xi^T P\xi \leq \alpha. \qquad (50)$$

The constants κ and α can e.g. be calculated as described in [6]. Using this, condition (4c) of Assumption 1 can be transformed into the following LMI analogously to above:

$$\begin{bmatrix} 2(1-\kappa)X + \rho(E_{null}E_{null}^T X + XE_{null}E_{null}^T) & \sqrt{2}(X\tilde{A}_{\bar{x}}^T + Y^T\tilde{B}_{\bar{x}}^T) & \sqrt{2}XQ^{1/2} & Y^T R^{1/2} \\ \sqrt{2}(\tilde{A}_{\bar{x}}X + \tilde{B}_{\bar{x}}Y) & X & 0 & 0 \\ \sqrt{2}Q^{1/2}X & 0 & I & 0 \\ R^{1/2}Y & 0 & 0 & I \end{bmatrix} \geq 0. \qquad (51)$$

As noted earlier, the point around which the linearized system dynamics are considered is not known a priori, but may vary in each time step. Hence a sufficient condition for (4c) to hold inside the terminal region $\mathcal{X}^f \cap \bar{\mathcal{X}}$ at each time step is that the LMI (51) as well as (50) hold for all $\bar{x} \in \bar{\mathcal{X}}$. As we assume the set $\bar{X}$ to be compact, and as f is assumed to be continuous, for each $\bar{x} \in \bar{X}$ the linearized system matrices $\tilde{A}_{\bar{x}}$ and $\tilde{B}_{\bar{x}}$ lie in some compact sets $\mathcal{A} \in \mathbb{R}^{nM \times nM}, \mathcal{B} \in \mathbb{R}^{nM \times mN}$. Furthermore, $\Psi_{\bar{x}}(\xi)$ lies in a compact set for all $\bar{x} \in \bar{X}$ if ξ lies in a compact set. Hence constants κ and α can be calculated such that (50) holds for all $\bar{x} \in \bar{X}$ (which, e.g., can be done as described in [6]), and ensuring that the LMI (51) holds for all $\bar{x} \in \bar{X}$ can be done by ensuring that it holds for the extreme points of the convex hull of $\mathcal{A} \times \mathcal{B}$ (or a polytopic outer approximation of it).

Remark 5 *Ensuring that the LMI* (51) *holds for all $\bar{x} \in \bar{X}$ might in general be quite conservative. A possible relaxation would be to allow for different, state-dependent, controller gains $K_{\bar{x}}$. However, the question then arises which controller gain each agent uses for computing its feasible trajectory which it sends to its neighbors, as the arithmetic mean of all agents $\bar{x}$ is unknown to each agent. One possibility would be that each agent uses K_{x_i}, i.e., the control gain according to its local state. However, a theoretical examination of such a modified scheme is beyond the scope of this paper.*

5.2.1 Example - synchronization of Van der Pol oscillators

Consider as an example four *Van der Pol oscillators* with dynamic equations

$$\dot{x}_i = f(x_i, u_i) := \begin{bmatrix} x_{i,2} + u_i \\ -x_{i,1} + \varepsilon(x_{i,2} - \frac{1}{3}x_{i,2}^3) \end{bmatrix}, \qquad i = 1, \dots, 4, \qquad (52)$$

where $\varepsilon > 0$ is a parameter specifying the degree of nonlinearity. The oscillators (52) exhibit a unique, exponentially stable limit cycle whose amplitude can be computed numerically (see,

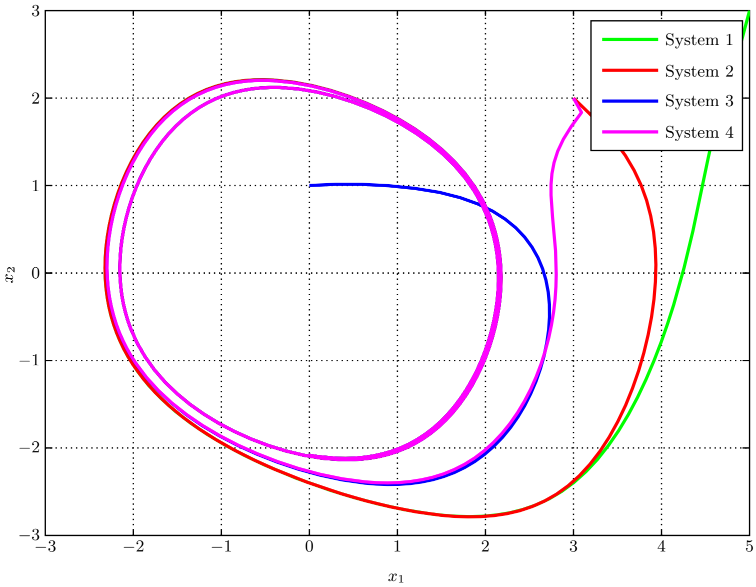


Figure 4: Synchronization of Van der Pol oscillators via DMPC in Section 5.2.1.

e.g., [28] and the references therein). The interconnection topology of the four oscillators is given by a line graph. Our goal is to synchronize the four oscillators via the DMPC algorithms developed in Section 3. To this end, we first derive a simple approximate model of the system (52) by standard Euler discretization with sampling time t^s:

$$x_i(k+1) = \begin{bmatrix} x_{i,1} + t^s x_{i,2} + t^s u_i \\ x_{i,2} - t^s x_{i,1} + t^s \varepsilon (x_{i,2} - \frac{1}{3} x_{i,2}^3) \end{bmatrix}. \tag{53}$$

Linearization of the system (53) around some point $(\bar{x}, 0)$ yields the system matrices

$$A_{\bar{x}} = \begin{bmatrix} 1 & t^s \\ -t^s & 1 + \varepsilon t^s (1 - \bar{x}_2^2) \end{bmatrix}, \qquad B_{\bar{x}} = \begin{bmatrix} t^s \\ 0 \end{bmatrix}. \tag{54}$$

The set $\bar{\mathcal{X}}$ is computed numerically as a compact convex set according to the above discussed requirements and such that it contains the unique limit cycle of the Van der Pol oscillators. From (54), one can see that ensuring that the LMI (51) holds for all $\bar{x} \in \bar{\mathcal{X}}$ reduces to ensuring it to hold for the two extreme matrices $A_{\bar{x}'}$ and $A_{\bar{x}''}$ resulting from $\bar{x}_2' = 0$ and $\bar{x}_2'' = \max_{\bar{x} \in \bar{\mathcal{X}}} |\bar{x}_2|$. The input constraints are such that $|u_i| \leq 5$, and the weighting matrices Q_i for the stage cost (31) are chosen such that $\xi_2 := x_2 - x_3$ is penalized ten times less than $\xi_1 := x_1 - x_2$ and $\xi_3 := x_3 - x_4$. Figure 4 shows simulation results for $\varepsilon = 0.43$, $t^s = 0.05$, and prediction horizon $T = 50$. One can see that the Van der Pol oscillators synchronize, and that indeed ξ_1 and ξ_3 converge faster to zero than ξ_2, as desired.

6 Conclusions

In this paper, we presented a general DMPC framework for cooperative control. We proposed two algorithms, one for general cost functions, and the second for a certain type of separable cost functions. We showed that both algorithms result in asymptotic stability of the closed-loop system with respect to the desired target set $\mathcal{X}^0$. Furthermore, we illustrated how several typical cooperative control problems such as the stabilization of an a priori known setpoint as well as consensus and synchronization tasks can be handled within this framework. In particular, we showed how the terminal region and the local auxiliary controllers can be calculated in these cases. Finally, we illustrated our results with several simulation examples.

The results presented in this paper were only obtained for nominal system dynamics, i.e., without considering disturbances. Hence future research will include designing robust distributed MPC schemes which are suited for cooperative control tasks. Furthermore, an important question to address is how the proposed algorithms perform in case of packet drops or delays in the communication between the systems.

References

[1] R. Olfati-Saber, J. A. Fax, and R. M. Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.

[2] A. Jadbabaie, J. Lin, and A. S. Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6):988–1001, 2003.

[3] R. Olfati-Saber and R. M. Murray. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Transactions on Automatic Control*, 49(9):1520–1533, 2004.

[4] D. W. Clarke, C. Mohtadi, and P. S. Tuffs. Generalized predictive control - Part I. The basic algorithm. *Automatica*, 23(2):137–148, 1987.

[5] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36:789–814, 2000.

[6] H. Chen and F. Allgöwer. A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability. *Automatica*, 34(10):1205–1217, 1998.

[7] R. Scattolini. Architectures for distributed and hierarchical model predictive control - a review. *Journal of Process Control*, 19:723–731, 2009.

[8] W. B. Dunbar and R. M. Murray. Distributed receding horizon control for multi-vehicle formation stabilization. *Automatica*, 42:549–558, 2006.

[9] T. Keviczky, F. Borrelli, and G. J. Balas. Decentralized receding horizon control for large scale dynamically decoupled systems. *Automatica*, 42:2105–2115, 2006.

[10] A. Richards and J. P. How. Robust distributed model predictive control. *International Journal of Control*, 80(9):1517–1531, 2007.

[11] C. Wang and C.-J. Ong. Distributed model predictive control of dynamically decoupled systems with coupled cost. *Automatica*, 46:2053–2058, 2010.

[12] J. Liu, D. Muñoz de la Peña, and P. D. Christofides. Distributed model predictive control of nonlinear systems subject to asynchronous and delayed measurements. *Automatica*, 46:52–61, 2010.

[13] B. T. Stewart, A. N. Venkat, J. B. Rawlings, S. J. Wright, and G. Pannocchia. Cooperative distributed model predictive control. *Systems & Control Letters*, 59:460–469, 2010.

[14] B. T. Stewart, S. J. Wright, and J. B. Rawlings. Cooperative distributed model predictive control for nonlinear systems. *Journal of Process Control*, 21:698–704, 2011.

[15] J. M. Maestre, D. Muñoz de la Peña, E. F. Camacho, and I. Alamo. Distributed model predictive control based on agent negotiation. *Journal of Process Control*, 21:685–697, 2011.

[16] P. Giselsson and A. Rantzer. Distributed model predictive control with suboptimality and stability guarantees. In *Proceedings of the 49th IEEE Conference on Decision and Control*, pages 7272–7277, 2010.

[17] M. D. Doan, T. Keviczky, and B. De Schutter. A dual decomposition-based optimization method with guaranteed primal feasibility for hierarchical MPC problems. In *Proceedings of the 18th IFAC World Congress*, pages 392–397, 2011.

[18] M. A. Müller, M. Reble, and F. Allgöwer. A general distributed MPC framework for cooperative control. In *Proceedings of the 18th IFAC World Congress*, pages 7987–7992, 2011.

[19] G. Ferrari-Trecate, L. Galbusera, M. P. E. Marciandi, and R. Scattolini. Model predictive control schemes for consensus in multi-agent systems with single- and double-integrator dynamics. *IEEE Transactions on Automatic Control*, 54(11):2560–2572, 2009.

[20] B. J. Johansson, A. Speranzon, M. Johansson, and K. H. Johansson. Distributed model predictive consensus. In *Proceedings of the 17th International Symposium on Mathematical Theory of Networks and Systems*, pages 2438–2444, 2006.

[21] T. Keviczky and K. H. Johansson. A study on distributed model predictive consensus. In *Proceedings of the 17th IFAC World Congress*, pages 1516–1521, 2008.

[22] J. B. Rawlings and D. Q. Mayne. *Model Predictive Control: Theory and Design.* Nob Hill Publishing, Madison, WI, 2009.

[23] C. Godsil and G. Royle. *Algebraic Graph Theory.* Springer, New York, 2001.

[24] M. V. Kothare, V. Balakrishnan, and M. Morari. Robust constrained model predictive control using linear matrix inequalities. *Automatica*, 32(10):1361–1379, 1996.

[25] D. Zelazo and M. Mesbahi. Edge agreement: Graph theoretic performance bounds and passivity analysis. *IEEE Transactions on Automatic Control*, 56(3):544–555, 2011.

[26] P. Finsler. Über das Vorkommen definiter und semi-definiter Formen in Scharen quadratischer Formen. *Comentarii Mathematici Helvetici*, 9:188–192, 1937.

[27] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory.* Society for Industrial and Applied Mathematics, 1994.

[28] J. Grasman, F. Verhulst, and S.-D. Shih. The Lyapunov exponents of the Van der Pol oscillator. *Mathematical Methods in the Applied Sciences*, 28(10):1131–1139, 2005.