Asadi, E. F., & Richards, A. (2018). Scalable distributed model predictive control for constrained systems. *Automatica*, *93*, 407-414. https://doi.org/10.1016/j.automatica.2018.03.050, https://doi.org/10.1016/j.automatica.2018.03.050

Scalable Distributed Model Predictive Control for Constrained Systems

Fatemeh Asadi, Arthur Richards

Aerospace Engineering Department, Queens Building, University of Bristol, Bristol, UK

Abstract

A distributed model predictive control strategy is proposed for subsystems coupled through their constraints. Self-organized Time Division Multiple Access is used to coordinate subsystem controllers in a sequence such that no two re-optimize simultaneously. This new approach requires no central coordination or pre-organized optimizing sequence. The scheme guarantees satisfaction of coupled constraints despite dynamic entry and exit of subsystems.

Key words: Distributed Model Predictive Control; Constrained Systems.

1 Introduction

Controlling large-scale systems such as transport networks or power distribution grids in a centralized way is often hard due to the computational scaling and coordination requirements. Centralized control is also prone to single points of failure, motivating interest in distributed control systems. Model Predictive Control (MPC) is a control technique combining constrained optimization with feedback control [Maciejowski, 2002, Grüne and Pannek, 2011], and schemes for Distributed MPC (DMPC) have been discussed by Scattolini [2009], Christofides et al. [2013], Negenborn and Maestre [2014] and many more. This paper focusses on DMPC for subsystems sharing a limited resource, which couples the systems through constraints [Keviczky et al., 2006, Kuwata et al., 2007a, Müller et al., 2012, Bourdais et al., 2014, Tedesco et al., 2014, Lucia et al., 2015, Li et al., 2016]. Other forms of coupling, not considered here, are through the dynamics [Dunbar, 2007, Alessio et al., 2011, Farina and Scattolini, 2012, Hernandez and Trodden, 2016] or through the system-wide objective function [Borrelli and Keviczky, 2006, Dunbar and Murray, 2006, Wang and Ong, 2010].

This paper adopts a *serial* DMPC scheme in which only one subsystem controller may optimize its plan at a time. With the plans for other subsystems therefore fixed, and known via communication, system-wide feasibility is this ensured. Previous work on serial schemes has proven its properties, subject to the assumption of an agreed updating sequence for the subsystems [Richards and How, 2007, Keviczky et al., 2004b,a, Kuwata et al., 2007a, Trodden and Richards, 2013, Dai et al., 2015]. However, the determination of that sequence is a centralized process. The simple contribution of this paper is the incorporation of a distributed slot allocation process, inspired by multiple access channel (MAC) sharing methods from communications systems [Rom and Sidi, 2012]. In particular, Self-organizing Time Division Multiple Access (STDMA) [Gaugel et al., 2013] is adopted, but instead of allocating *transmission slots* for communication, here it allocates *optimization slots* for re-planning.

The goal of distributed sequencing (slot allocation) is the same in spirit with the goal of "Plug & Play (PnP) Control" which intends to handle the distributed control problem for systems with a changing numbers of subsystems. In PnP control by adding or removing a subsystem, just local controller of the subsystem under control and subsystems influenced by it (neighbours) need to be redesigned [Stoustrup, 2009]. PnP control methods proposed by Riverso et al. [2014] and Zeilinger et al. [2013] tackle the problem of automatically accommodating the constant changes in system model due to adding or removing one or multiple subsystems during closed-loop operation. In both of these works, subsystems are physically coupled whereas in this paper subsystems are coupled through their constraints. Barreiro-Gomez et al. [2015] and Lucia et al. [2015] addressed the challenge of performing network changes because of joining and leaving subsystems with coupled constraints. The de-

Email addresses: elham.asadi@bristol.ac.uk (Fatemeh Asadi), arthur.richards@bristol.ac.uk (Arthur Richards).

centralized MPC scheme presented by Barreiro-Gomez et al. [2015] can handle only one single coupled constraint on control signals.The contract-based DMPC introduced by Lucia et al. [2015] guarantees the constraint satisfaction in parallel optimization via transmission of sequences of possible future trajectories. The proposed method in this paper considers the particular coupling constraints associated with sharing of limited resources. Also in contrast to Lucia et al. [2015], our subsystems communicate exact trajectories but with serial (one-at-a-time) optimization and they implement a decentralized approach to sequencing.

2 Self-Organized Sequencing

Consider a dynamic set of subsystems $\mathcal{P}(k)$ containing $n(k) = |\mathcal{P}(k)|$ members at each time step k. A subset of these subsystems $\mathcal{P}_C(k) \subseteq \mathcal{P}(k)$ are in the *cooperation mode*, using serial DMPC to coordinate their actions such that shared resource limits are respected. Let $n_C(k) = |\mathcal{P}_C(k)|$ denote the number of cooperating subsystems. The remainder $\mathcal{P}(k) \setminus \mathcal{P}_C(k)$ remain in a restricted *safe mode*, not consuming any of the shared resources, and hence not required to communicate. Serial DMPC requires a unique allocation of subsystems to time steps, such that every step is associated to at most one subsystem, $p_k \in \mathcal{P}_C(k) \cup \{0\}$. At every step, the allocated subsystem p_k (if there is one, $p_k \neq 0$) solves its local optimal control problem and shares the resulting intentions with the others. This section describes how the allocation is achieved in a dynamic, self-organized way, enabling subsystems to move from safe mode to cooperation mode. Since this problem is analogous to slot allocation in communications, the algorithm is based on STDMA, which is a decentralized MAC method.

Define $L_f \geq 1$ to be the *frame length*, *i.e.* the repeating period for slot allocation, such that $p_k = p_{k+L_f}$ provided p_k remains in cooperation. Then Algorithm 1 presents the procedure of self-organized sequencing for an agent $q \in \mathcal{P}(k) \setminus \mathcal{P}_C(k)$ wishing to enter cooperation mode. Since the allocation is periodic, entry involves simply listening for one frame and then choosing an available slot in the next frame. The possible problem is a "collision" in which two subsystems attempt to enter at the same step, each unaware of the presence of the other. This event is detected by both subsystems and a random back-off time is employed to avoid deadlock.

Leaving the cooperation is achieved by stopping transmission, indicating to others that the slot is again available. Thus, unlike the communications case where slot allocations have finite lifetime, a slot belongs to a subsystem indefinitely until that subsystem relinquishes it. The control constraints associated with entry and leaving are described in Section 4.

Assumption 1 (Frame Length) *The frame length L_f is known to all subsystem controllers. This forms part of the common interface for subsystems: it is central to the scalability concept that the interface is standard and known to all agents.*

Algorithm 1 Entry into Cooperation Mode

Require: Subsystem ID q, initial time k_1
1: Listen for L_f steps to determine $\{p_{k_1}, \ldots, p_{k_1+L_f}\}$
2: Identify offsets of free slots: $\mathcal{J}_{free} = \{j \in [0, \ldots, L_f] \mid p_{k_1+j} = 0\}$
3: **if** no free slot, $\mathcal{J}_{free} = \emptyset$ **then**
4: Try again: go to Step 1
5: **else**
6: Choose free slot at random, $\hat{j} \in \mathcal{J}_{free}$
7: Wait for slot $\hat{j}$ in next frame, $k = k_1 + L_f + \hat{j}$
8: Transmit current plan $Y_p^*(k_1 + L_f + \hat{j})$
9: **if** no other subsystem transmitted **then**
10: Secured $p_{k_1+\hat{j}+nL_f} = q \ \forall n = 1, 2, \ldots$ as long as $q \in \mathcal{P}_C(k)$
11: **return** Success
12: **else**
13: Collision: wait for random number of steps
14: Try again: go to Step 1
15: **end if**
16: **end if**

Remark 1 *It is not necessary for all subsystem controllers to define a common phasing of the frames, since the frames are periodic [Rom and Sidi, 2012].*

Remark 2 *Since no more than L_f subsystems can have slots, then the frame length L_f represents a limit on the number of subsystems in cooperation mode: $n_C(k) \leq L_f$. The choice of L_f therefore represents an important design choice, as increasing L_f means more capacity for entering agents but a longer wait to enter, according to Algorithm 1. A full study of this trade-off is beyond the scope of this brief paper and the reader is directed to Asadi and Richards [2015] for more consideration.*

3 Control Problem Definition

Each subsystem $p \in \mathcal{P}(k)$ has its own dynamics,

$$\mathbf{x}_p(k+1) = f_p(\mathbf{x}_p(k), \mathbf{u}_p(k)) \quad k \in \mathbb{N}, \quad \forall p \in \mathcal{P} \quad (1)$$

where $\mathbf{x}_p \in \mathbb{R}^{N_x,p}$ and $\mathbf{u}_p \in \mathbb{R}^{N_u,p}$ are the state vector and control input vector of subsystem p, respectively.

Remark 3 *The dynamics (1) are not subject to any uncertainty. Ideas such as the tube approach [Trodden and Richards, 2010] could be applied to handle disturbances, but these are omitted here for simplicity.*

Each subsystem p is subject to local constraints on state and input

$$\mathbf{x}_p(k) \in \mathcal{X}_p \quad (2)$$

$$\mathbf{u}_p(k) \in \mathcal{U}_p \tag{3}$$

and has its own local objective function, in fixed horizon MPC form

$$J_p = \sum_{t=0}^{N-1} l_p(\mathbf{x}_p(k+t|k), \mathbf{u}_p(k+t|k), k) + V_p(\mathbf{x}_p(k+N|k), k) \tag{4}$$

where N is the number of time steps in the prediction horizon, $l_p : \mathbb{R}^{N_x,p} \times \mathbb{R}^{N_u,p} \times \mathbb{R} \to \mathbb{R}_{0+}$ represents a stage cost, V_p is a terminal cost and the double subscript notation $(k+t|k)$ indicates the prediction of a variable t steps ahead from time k. The focus of this paper is constraint satisfaction, and hence the nature of the cost function will not be specified in more detail. However, the reader should note that the time variation permits different costs to be used in different modes, and this will be exploited later.

Define a set of shared resources $\mathcal{L}$ and let $y_p^\ell \in \mathbb{R}$ be the amount of particular resource $\ell \in \mathcal{L}$ by subsystem p:

$$y_p^\ell(k) = g_p^\ell(\mathbf{x}_p(k), \mathbf{u}_p(k)) \tag{5}$$

All subsystems share the limited resources $\mathcal{L}$ and hence their outputs are coupled by

$$\sum_{p=1}^{n} y_p^\ell(k) \leq 1, \ \forall \ell \in \mathcal{L}. \tag{6}$$

Note each resource is scaled to have exactly one unit of resource. Define $\mathbf{y}_p(k)$ as the combination of all resource usage outputs.

$$\mathbf{y}_p(k) := \{y_p^\ell(k)\}_{\ell \in \mathcal{L}}$$

This notation is used to enable the set of resources to be continuous, as will appear in forthcoming examples. In the case of a simple discrete set of resources, $\mathbf{y}$ will simply be a vector. Typical vector notation will be employed in the sequel, with $\mathbf{y}_1 = \mathbf{y}_2$ implying $y_1^\ell = y_2^\ell \ \forall \ell \in \mathcal{L}$ and $\mathbf{y}_1 \leq \mathbf{y}_2$ implying $y_1^\ell \leq y_2^\ell \ \forall \ell \in \mathcal{L}$. Also define $\mathcal{C}_n$ as the set of feasible outputs for n subsystems:

$$\mathcal{C}_n := \left\{ (\mathbf{y}_1, \ldots, \mathbf{y}_n) \mid \sum_{p=1}^{n} \mathbf{y}_p \leq \mathbf{1} \right\}. \tag{7}$$

The slot allocation process in Section 2 ensures that every time step k is uniquely allocated to one subsystem $p_k \in \mathcal{P}_C \cup \{0\}$. A later section will show how this ensures constraint satisfaction. However, because the allocation process takes some time, it is necessary to consider the case of subsystems that have entered the problem but not yet secured a slot, and hence not been able to coordinate their plans with other agents. For this reason, a *safe mode* of operation is required for systems entering (and leaving) where subsystems can not use the shared resource. This mode requires the existence of an invariant set with resource outputs equal to zero.

Proposition 1 *Feasibility of Safe Mode*

$$\mathcal{C}_n \times \mathbf{0}^m \subseteq \mathcal{C}_{n+m} \ \forall m, n \tag{8}$$

The proof is trivial: adding zero does not change the outputs, so if the first n satisfy the upper limit, so must all $n+m$.

As is common for MPC, recursive feasibility will depend on invariant set constraints. Define $\mathcal{F}_p(\mathcal{Y}_p)$ to be an invariant set of states for subsystem p satisfying output set constraints $\mathcal{Y}_p$:

$$\forall \mathbf{x}_p \in \mathcal{F}_p(\mathcal{Y}_p) \ \exists \kappa_p(\mathbf{x}_p) \in \mathcal{U}_p \ : \tag{9}$$

$$f_p(\mathbf{x}_p, \kappa_p(\mathbf{x}_p)) \in \mathcal{F}_p(\mathcal{Y}_p) \tag{10}$$

$$\mathbf{y}_p(\mathbf{x}_p, \kappa_p(\mathbf{x}_p)) \in \mathcal{Y}_p \tag{11}$$

The choice of these sets $\mathcal{F}_p$ is left to the designer. Maximal invariant sets [**?**] or simple equilibria conditions can be employed.

Assumption 2 (Safe Mode Invariance) *An invariant set exists satisfying the conditions of safe mode, $\mathcal{F}_p(\{\mathbf{0}\}) \neq \emptyset$.*

Example 1 (Power outlets) *For a set of power outlets with total output power limited to $P_{\max}$, forming just one shared resource $\mathcal{L} = \{1\}$, and define a single coordinating output y_p^1 as the proportion of available power drawn by each connected subsystem:*

$$g_p^1(\mathbf{x}_p(k), \mathbf{u}_p(k)) = \frac{P_p(\mathbf{x}_p(k), \mathbf{u}_p(k))}{P_{\max}}$$

where $P_p(\mathbf{x}_p(k), \mathbf{u}_p(k))$ is the power consumed by subsystem p dependent on its dynamic state and input.

Example 2 (Air Traffic) *Consider a set of aircraft flying at the same altitude within an airspace sector $\mathcal{A} \subseteq \mathbb{R}^2$. Whilst inside the sector, aircraft are required to maintain a minimum separation $R_{\min}$. (Other sector controllers, not considered here, provide separation outside $\mathcal{A}$.) Each point in the airspace is considered a shared resource, and separation is assured if no more than one aircraft comes within $\frac{R_{\min}}{2}$ of each point. Hence the resource set $\mathcal{L} = \mathcal{A}$ and the coordinating output is*

$$g_p^\ell = \begin{cases} 1 & \|\ell - \mathbf{r}_p(\mathbf{x}_p)\| < \frac{R_{min}}{2} \wedge \mathbf{r}_p(\mathbf{x}_p) \in \mathcal{A} \\ 0 & otherwise \end{cases}$$

where $\mathbf{r}_p(\mathbf{x}_p) \in \mathbb{R}^2$ represents the position of aircraft p. Safe mode is equivalent to $\mathbf{r}_p(\mathbf{x}_p) \notin \mathcal{A}$. For a fixed-wing aircraft, circling flight at safe separation from all occupied air space provides a suitable invariant set $\mathcal{F}(\mathcal{Y})$ [Kuwata et al., 2007b].

4 Scalable DMPC

4.1 Overview

The 'lifecycle' of a subsystem q over a single interaction with the control system is as follows:

1. *Inactive*: q is not part of the system and $q \notin \mathcal{P}(k)$
2. *Safe*: $q \in \mathcal{P}(k) \setminus \mathcal{P}_C(k)$ *i.e.* active but not yet co-operating. q constrained such that $\mathbf{y}_q(k) = \mathbf{0}$. Algorithm 1 followed until slots $p_k = q$ are secured.
3. *Cooperation*: $q \in \mathcal{P}_C(k)$ and communication is used to maintain system-wide feasibility
4. *Leaving*: $q \in \mathcal{P}_C(k)$ and communication on-going. q moves back to safe mode $\mathbf{y}_q(k) \to \mathbf{0}$
5. *Safe*: $q \in \mathcal{P}(k) \setminus \mathcal{P}_C(k)$ *i.e.* active but no longer co-operating. q constrained such that $\mathbf{y}_q(k) = \mathbf{0}$
6. *Inactive*: q is not part of the system and $q \notin \mathcal{P}(k)$.

Remark 4 *Note that there is no need to an "Entering" mode as counterpart to "Leaving". This is because all trajectories in safe mode satisfy the requirements of cooperation mode, so the transition from safe to cooperation can happen without waiting, as soon as a slot has been secured. However, the converse is not true, in that not all feasible cooperating trajectories are in safe mode, so a transition stage is required to go from cooperation to safe. This transition is the "Leaving" mode.*

Assumption 3 (Mode change timing) *Each subsystem is equipped with some way to decide when to enter and leave co-operation.*

Remark 5 *Mode change timing decisions are not part of the control protocol itself. For the power example, entering might be triggered by a command, and leaving by a certain period of zero consumption. For air traffic, entering and leaving would be triggered by location relative to the boundary of the sector. For the remainder, all subsystems are assumed able to enter (move to stage 2) or leave (move to stage 5) at any time.*

4.2 MPC Optimization

Different types of constraints are applied for each subsystem depending on its mode of operation. Fortunately, the same formulation of MPC can be used for all of them. The MPC problem depends on both the initial subsystem state $\mathbf{x}_p$ and a set of output constraints $\widetilde{\mathcal{Y}}_p(k) = \mathcal{Y}_p(k|k) \times \mathcal{Y}_p(k+1|k) \times ... \times \mathcal{Y}_p(k+N|k)$ dependent on the mode of operation and communications from cooperating subsystems. It optimizes a sequence of future controls $U_p(k) = (\mathbf{u}_p(k|k), \mathbf{u}_p(k+1|k), ..., \mathbf{u}_p(k+N-1|k))$ with the general set of constraints for each subsystem p is defined as:

$$\mathbb{P}_p(\mathbf{x}_p(k), \widetilde{\mathcal{Y}}_p(k)) : \min_{U_p(k)} \; J_p \tag{12a}$$

subject to $\forall t \in \{0, ..., N-1\}$:

$$\mathbf{x}_p(k|k) = \mathbf{x}_p(k) \tag{12b}$$
$$\mathbf{x}_p(k+t+1|k) = f_p(\mathbf{x}_p(k+t|k), \mathbf{u}_p(k+t|k)) \tag{12c}$$
$$\mathbf{y}_p(k+t|k) = g_p(\mathbf{x}_p(k+t|k), \mathbf{u}_p(k+t|k)) \tag{12d}$$
$$\mathbf{x}_p(k+t|k) \in \mathcal{X}_p \tag{12e}$$
$$\mathbf{u}_p(k+t|k) \in \mathcal{U}_p \tag{12f}$$
$$\mathbf{y}_p(k+t|k) \in \mathcal{Y}_p(k+t|k) \tag{12g}$$
$$\mathbf{x}_p(k+N|k) \in \mathcal{F}_p(\mathcal{Y}_p(k+N|k)) \tag{12h}$$

For convenience we also define output *sequences*:

$$Y_p(k) = (\mathbf{y}_p(k|k), \mathbf{y}_p(k+1|k), ..., \mathbf{y}_p(k+N|k)) \tag{13}$$

and note that $Y_p(k) \in \widetilde{\mathcal{Y}}(k)$ is equivalent to $\mathbf{y}_p(k+t|k) \in \mathcal{Y}_p(k+t|k)\ \forall t \in \{0, \ldots, N\}$. Note that these constraints include the terminal step output $t = N$ although this is not explicitly formed in the optimization. The handling of the terminal output will be explained later in Section 4.6. It is these output sequences that will be communicated between subsystems when in cooperation mode.

4.3 Constraints for Safe Mode

Safe mode requires $\mathbf{y}_p(k) = \mathbf{0}$, decoupled from other subsystems, so the constraint required is $\widetilde{\mathcal{Y}}_p(k) = \{\mathbf{0}\}^{N+1}$ since

$$Y_p(k) \in \mathbf{0}^{N+1} \Leftrightarrow \mathbf{y}_p(k+t|k) = \mathbf{0}\ \forall t \in \{0, ..., N\} \tag{14}$$

4.4 Constraints for Entering and Cooperation Mode

Define $\mathcal{P}_{\{-p\}}(k)$ as the set of neighbours of the subsystem p at time instance k, defined as those subsystems who are cooperating with p, $\mathcal{P}_{\{-p\}}(k) := \mathcal{P}_C(k) \setminus p = \{q_{p,1}, \ldots, q_{p,n_C(k)-1}\}$. Note that the definition of neighbours is only used if $p \in \mathcal{P}_C(k)$. Then the constraints on the outputs of a subsystem p when in cooperation mode are $\widetilde{\mathcal{Y}}_p(k) = \mathsf{C}_p(Y_{\{-p\}}(k))$ where $Y_{\{-p\}}(k) = \{Y_q(k)\}_{q \in \mathcal{P}_{\{-p\}}(k)}$ is the collection of output sequences for all neighbours of p, shared via communication, and

$$\mathsf{C}_p(Y_{\{-p\}}) := \Big\{ Y_p \;\Big|\; (\mathbf{y}_p(k+t|k), \mathbf{y}_{q_{p,1}}(k+t|k), ..., \mathbf{y}_{q_{p,n_C(k)-1}}(k+t|k)) \in \mathcal{C}_{n_C(k)} \forall t \in \{0, \ldots, N\} \Big\} \tag{15}$$

Given the form of the coupling constraints (7), this is equivalent to

$$\mathsf{C}_p(Y_{\{-p\}}) := \Big\{ Y_p \in \mathbb{R}^{N_y(N+1)} \;\Big|\; \mathbf{y}_p(k+t|k) \leq \mathbf{1} - \sum_{q \in \mathcal{P}_{\{-p\}}} \mathbf{y}_q(k+t|k) \forall t \in \{0, \ldots, N\} \Big\} \tag{16}$$

Also define $\bar{\mathsf{C}}_n$ to be the set of all combinations of n feasible output sequences:

$$\bar{\mathsf{C}}_{n_C(k)} := \Big\{ \{Y_p\}_{p \in \mathcal{P}_C(k)} \in \mathbb{R}^{n N_y(N+1)} \;\Big|$$

$$(y_1(k+t|k), ..., y_{n_C(k)}(k+t|k)) \in \mathcal{C}_{n_C(k)} \forall t \in \{0, ..., N\}\Big\} \quad (17)$$

Then by construction

$$\{Y_p\}_{p\in\mathcal{P}_C(k)} \in \bar{\mathsf{C}}_{n_C(k)} \Leftrightarrow Y_q(k) \in \mathsf{C}_q(Y_{\{-q\}}(k)) \; \forall q \in \mathcal{P}_C(k) \quad (18)$$

4.5 Constraints for Leaving Mode

When leaving cooperation mode, the terminal state is constrained to satisfy the requirements of safe mode $\mathbf{y}_p(k+N|k) = \mathbf{0}$, such that the subsystem state transitions to satisfy safe mode, $\mathbf{y}_p(k) \to \mathbf{0}$. However it must ensure feasibility with respect to neighbours before it reaches zero output by considering published plans. The constraints for leaving mode are therefore given by an additional restriction on those for cooperation mode:

$$\mathsf{L}_p(Y_{\{-p\}}) := \mathsf{C}_p(Y_{\{-p\}}) \cap \{\mathbb{R}^{N_y N} \times \{\mathbf{0}\}\} \quad (19)$$

Assumption 4 (Convergence of leaving mode) *If $\mathbb{P}_p(\mathbf{x}_p(k), \mathsf{L}_p(Y_{\{-p\}}))$ is recursively feasible then $\mathbf{y}_p(k) \to \mathbf{0}$. This relies on suitable construction of the cost terms in (4) and standard MPC results [Mayne et al., 2000]. Note that the cost is permitted to be time-varying, hence it is acceptable for a different cost to be used in leaving mode than in other modes, and this assumption only affects the cost employed when in leaving mode.*

4.6 Communicated Output Sequence

At time step k, the allocated subsystem p_k (if there is one) solves its optimization and communicates its output sequence (13). Terms $(\mathbf{y}_p(k|k), \ldots, \mathbf{y}_p(k+N-1|k))$ are immediately available through constraints (12d). The final term is used to convey the maximum resource levels anticipated to be consumed at any subsequent step under the invariance control law:

$$\mathbf{y}_p(k+N|k) := \max_{t\geq N} \mathbf{g}_p(\mathbf{x}_p(k+t|k), \kappa_p(\mathbf{x}_p(k+t|k))) \quad (20)$$

where the max is performed elementwise and $\forall t \geq N$:

$$\mathbf{x}_p(k+t+1|k) = \mathbf{f}_p(\mathbf{x}_p(k+t|k), \kappa_p(\mathbf{x}_p(k+t|k)))$$

By the construction in (10) of the invariant set $\mathcal{F}_p(\mathcal{Y}_p)$ employed in the terminal constraint (12h), it is known that $\mathbf{y}_p(k+N|k) \in \mathcal{Y}_p(k+N|k)$ and hence must be finite.

4.7 Construction of Tail Solution

It is common in constrained MPC for the 'tail solution', a continuation of the control sequence from the preceding time step, to play a role in proof of feasibility [Mayne et al., 2000]. In serial DMPC, the tail is used explicitly by some subsystems if it is not their turn in the sequence $\{p_k\}$. This can be viewed as an extreme form of contract-based DMPC [Lucia et al., 2015] in which the contract is reduced to a single sequence of controls. For the MPC optimal control problem in (12) the tail is constructed as follows:

$$\hat{U}_p(U_p^*(k-1)) = (\mathbf{u}_p^*(k|k-1), ..., \mathbf{u}_p^*(k+N-2|k-1), \kappa_p(\mathbf{x}_p^*(k-1+N|k-1))) \quad (21)$$

where the superscript * denotes the result from the preceding time step, either from optimization or from the preceding tail. This is bounded by the following output tail sequence

$$\hat{Y}_p(Y_p(k-1)) = (\mathbf{y}_p(k|k-1), ..., \mathbf{y}_p(k+N-2|k-1), g_p(\mathbf{x}_p(k+N-1|k-1), \kappa_p(\mathbf{x}_p(k+N-1|k-1)))) \quad (22)$$

Remark 6 *According to the conditions (9), (10) and (11), the state sequence ends at a control invariant set satisfying output constraints. This allows subsystems to predict intentions of their neighbours.*

4.8 Scalable DMPC Algorithm

Algorithm 2 presents the proposed scalable DMPC method in full, drawing on the definitions above. It covers stages 2-5 of the lifecycle described in Section 4.1: stages 1 and 6 are omitted as inactive subsystems do nothing, if they even exist. Algorithm 2 also encodes the basic 'rules' of the distributed control scheme:

- A subsystem is restricted to safe mode $\mathbf{y}_p \in \{\mathbf{0}\}$ unless it is cooperating with others, sharing its intentions Y_p using its allocated time slots.
  - A subsystem may not leave safe mode before it has secured an allocated slot.
  - A subsystem may not cease cooperation and relinquish its allocated slot before returning to safe mode.
- A cooperating subsystem is restricted to its tail sequence unless the current time step is its allocated slot.

In the following, * denotes the *adopted* controls or inputs, which may be determined either by optimization or use of the tail according to Algorithm 2 and communicated between subsystems.

Recursive feasibility is considered in the following section. Note however that it is not possible to guarantee feasibility *a priori* of the optimization for leaving mode $\mathbb{P}_q(\mathbf{x}_q(k), \mathsf{L}_q(Y^*_{\{-q\}}(k)))$ since its constraints are tighter than those for cooperation mode,

$$\mathsf{L}_q(Y^*_{\{-q\}}(k)) \subseteq \mathsf{C}_q(Y^*_{\{-q\}}(k)), \quad (23)$$

according to (19). Therefore the algorithm checks for feasibility of leaving mode and reverts to the cooperation mode problem in case of infeasibility.THis does raise the prospect of deadlock: there is no guarantee of being able to leave.

Algorithm 2 Updating plan of subsystem q at time k

Require: Slot allocations $\{p_k\}$,
neighbours' plans $Y^*_{\{-q\}}(k-1)$,
mode of q { safe, entering, cooperation, leaving }

Determine state $\mathbf{x}_q(k)$
if mode of q is 'safe' **then**
  Solve $\mathbb{P}_q(\mathbf{x}_q(k), \{\mathbf{0}\}^{N+1})$
  if q wants 'enter' the 'cooperation' **then**
    Follow next step of Algorithm 1
    if Algorithm 1 returned Success **then**
      Set mode of q to 'cooperation'
    end if
  end if
else
  if $p_{k-1} \neq 0$ **then**
    Receive neighbour update $Y^*_{p_{k-1}}(k-1)$
  end if
  Update neighbours' plans $Y^*_{\{-q\}}(k)$ using tails:
  $Y^*_p(k) \leftarrow \hat{Y}(Y^*_p(k-1))\ \forall p \in \mathcal{P}_{\{-q\}}(k-1)$
  if $p_k \neq q$ **then** ▷ Slot does not belong q
    Get own plan from tail $U^*_q(k) = \hat{U}(U^*_q(k-1))$
  else ▷ turn of q to optimize
    if mode of q is 'entering' or 'cooperation' **then**
      Solve $\mathbb{P}_q(\mathbf{x}_q(k), \mathsf{C}_q(Y^*_{\{-q\}}(k)))$
      Transmit $Y^*_q(k)$ to neighbours $\mathcal{P}_{\{-q\}}(k)$
    else ▷ Mode of q must be 'leaving'
      Try to solve $\mathbb{P}_q(\mathbf{x}_q(k), \mathsf{L}_q(Y^*_{\{-q\}}(k)))$
      if feasible **then**
        if $Y^*_q(k) \in \{\mathbf{0}\}$ **then**$^{N+1}$ and $\mathbf{x}^*_q(k+N|k) \in \mathcal{F}_q(\{\mathbf{0}\}$
          Set mode of q to 'safe'
          ▷ No transmission: giving up slot
        else ▷ Keep slot until back to safety
          Transmit $Y^*_q(k)$ to $\mathcal{P}_{\{-q\}}(k)$
        end if
      else ▷ Leaving opt. was infeasible
        Solve $\mathbb{P}_q(\mathbf{x}_q(k), \mathsf{C}_q(Y^*_{\{-q\}}(k)))$
        Transmit $Y^*_q(k)$ to $\mathcal{P}_{\{-q\}}(k)$
      end if
    end if
  end if
end if
Apply control $\mathbf{u}_q(k) = \mathbf{u}^*_q(k|k)$

5 Feasibility of Scalable DMPC

At the system-wide scale, the progression of the algorithm at each time step can be divided into four significant cases:

1. $p_k = 0$ and no subsystem enters
2. $p_k = 0$ and a subsystem uses the opportunity to enter cooperation
3. $p_k \neq 0$ and subsystem p_k re-optimizes and remains in cooperation
4. $p_k \neq 0$ and subsystem p_k re-optimizes and leaves cooperation

In parallel, any number of subsystems may remain in safe mode, or enter safe mode from an inactive state, or leave safe mode to become inactive. Note that if two or more subsystems attempt to enter at the same empty slot p_k, this is detected as a collision according to Algorithm 1, and none of them can enter. This section considers recursive feasibility and constraint satisfaction of the whole DMPC scheme. Stability is beyond the scope of this paper. The proof will be based on recursion, starting with a common standing assumption.

Assumption 5 *At time k_0, all subsystems have feasible plans, with $\{Y^*_p(k_0)\}_{p\in\mathcal{P}_C(k_0)} \in \bar{\mathsf{C}}_{n_C(k_0)}$ and $Y^*_p(k_0) \in \{\mathbf{0}\}^{N+1}\ \forall p \in \mathcal{P}(k_0) \setminus \mathcal{P}_C(k_0)$, and control sequences $U^*_p(k_0)$ satisfying the constraints of their local optimizations $\mathbb{P}_p(\mathbf{x}_p(k_0), \cdot)$.*

Proposition 2 (System-wide feasibility at k_0) *The combination of all subsystems, in safe, entering, cooperation and leaving modes, satisfy*

$$\{Y^*_p\}_{p\in\mathcal{P}(k_0)} \in \bar{\mathsf{C}}_{n(k_0)} \tag{24}$$

Proof: Note that the constraints on entering mode is same as cooperation mode and the constraints on leaving mode are tighter than those on cooperation mode according to (23). So, subsystems in entering and leaving modes can be assumed to satisfy constraints on cooperation mode. Then Assumption 5 implies $\forall t \in \{0, ..., N\}$:

$$(\mathbf{y}^*_1(k_0+t|k_0), ..., \mathbf{y}^*_{n_C(k_0)}(k_0+t|k_0)) \in \mathcal{C}_{n_C(k_0)} \tag{25}$$

and $\mathbf{y}^*_p(k_0+t|k_0) \in \{\mathbf{0}\}\ \forall p \in \mathcal{P}(k) \setminus \mathcal{P}_C(k)$ so by the requirements for safe mode (8) it follows that $\forall t \in \{0, ..., N\}$: $(\mathbf{y}^*_1(k_0+t|k_0), ..., \mathbf{y}^*_{n(k_0)}(k_0+t|k_0)) \in \mathcal{C}_{n(k_0)}$ which is equivalent to (24). □

Proposition 3 (Recursive feasibility in safe mode) *For any subsystem in safe mode $p \in \mathcal{P}(k) \setminus \mathcal{P}_C(k)$, optimization $\mathbb{P}_p(\mathbf{x}_p(k_0+1), \{\mathbf{0}\}^{N+1})$ is feasible.*

Proof: Safe mode is standard MPC with invariant terminal conditions and constant constraints, according to (14). The tail solution (21) is feasible ensuring feasibility of the optimization. □

Proposition 4 (Case 1: no update or entry) *If $p_{k_0} = 0$ and $\mathcal{P}_C(k_0+1) = \mathcal{P}_C(k_0)$, then*

$$\{Y^*_p(k_0+1)\}_{p\in\mathcal{P}_C(k_0+1)} \in \bar{\mathsf{C}}_{n_C(k_0+1)} \tag{26}$$

and

$$Y^*_p(k_0+1) \in \{\mathbf{0}\}^{N+1}\ \forall p \in \mathcal{P}(k_0+1) \setminus \mathcal{P}_C(k_0+1) \tag{27}$$

Proof: In this case, all cooperating subsystems adopt their tail solutions giving $Y^*_p(k_0+1) = \hat{Y}(Y^*_p(k_0))\ \forall p \in$

$\mathcal{P}_C(k_0+1)$ and hence

$$(\mathbf{y}_1^*(k_0+t|k_0+1),...,\mathbf{y}_{n_C(k_0)}^*(k_0+t|k_0+1)) = (\mathbf{y}_1^*(k_0+t|k_0),...,\mathbf{y}_{n_C(k_0)}^*(k_0+t|k_0)), \quad t=1..N \tag{28a}$$

$$(\mathbf{y}_1^*(k_0+N+1|k_0+1),...,\mathbf{y}_{n_C(k_0)}^*(k_0+N+1|k_0+1)) = (g_1(\mathbf{x}_1^*(k_0+N|k_0),\kappa_1(\mathbf{x}_1^*(k_0+N|k_0))),..., g_{n_C(k_0)}(\mathbf{x}_{n_C(k_0)}^*(k_0+N|k_0),\kappa_{n_C(k_0)}(\mathbf{x}_{n_C(k_0)}^*(k_0+N|k_0)))) \tag{28b}$$

and (25) shows each of those combinations to be feasible. (27) follows from Proposition 3. □

Proposition 5 (Case 2: subsystem entry) *If $p_{k_0} = 0$ and a new subsystem enters $\mathcal{P}_C(k_0+1) = \mathcal{P}_C(k_0)\cup\{q\}$, then*

$$\{Y_p^*(k_0+1)\}_{p\in\mathcal{P}_C(k_0+1)} \in \bar{\mathsf{C}}_{n_C(k_0+1)} \tag{29}$$

and

$$Y_p^*(k_0+1) \in \{\mathbf{0}\}^{N+1}\ \forall p \in \mathcal{P}(k_0+1)\setminus\mathcal{P}_C(k_0+1) \tag{30}$$

Proof: Since the subsystems already in cooperation will adopt their tail solutions, it follows from Proposition 4 that $\{Y_p^*(k_0+1)\}_{p\in\mathcal{P}_C(k_0)} \in \bar{\mathsf{C}}_{n_C(k_0)}$ *i.e.* that the adopted plans of those subsystems remain mutually feasible. Furthermore, the transmitted plan of (step 8 of Algorithm 1) must satisfy $Y_q^*(k_0+1) \in \mathsf{C}_q(Y_{\{-q\}}(k_0+1))$. Since q can only be entering from safe mode, $Y_q^*(k_0) \in \{\mathbf{0}\}^{N+1}$. So, the tail solution of q is $\hat{Y}(Y_q^*(k_0)) \in \{\mathbf{0}\}^{N+1}$ which satisfies $Y_q^*(k_0+1) \in \mathsf{C}_q(Y_{\{-q\}}(k_0+1))$. Thus following the same reasoning as Proposition 2 gives $(Y_q^*(k_0+1),\{Y_p^*(k_0+1)\}_{p\in\mathcal{P}_C(k_0)}) \in \bar{\mathsf{C}}_{n_C(k_0)+1}$ which is equivalent to (29). (30) for the subsystems remaining in safe mode follows from Proposition 3. □

Proposition 6 (Case 3: subsystem update) *If $p_{k_0} = q \neq 0$ then $\mathbb{P}_q(\mathbf{x}_q(k_0+1), \mathsf{C}_q(Y_{\{-q\}}^*(k_0+1)))$ is feasible and the result satisfies*

$$\{Y_p^*(k_0+1)\}_{p\in\mathcal{P}_C(k_0+1)} \in \bar{\mathsf{C}}_{n_C(k_0+1)} \tag{31}$$

and

$$Y_p^*(k_0+1) \in \{\mathbf{0}\}^{N+1}\ \forall p \in \mathcal{P}(k_0+1)\setminus\mathcal{P}_C(k_0+1) \tag{32}$$

Proof: Proposition 4 has shown that the tail solutions satisfy the system-wide constraints $\bar{\mathsf{C}}_{n_C(k_0+1)}$ and from (18) it follows that $\hat{Y}(Y_q^*(k_0)) \in \mathsf{C}_q(Y_{\{-q\}}^*(k_0+1))$ since $Y_{\{-q\}}^*(k_0+1)$ must also be constructed from the relevant tail solutions. Therefore $\hat{Y}(Y_q^*(k_0))$ is known *a priori* to be a solution to $\mathbb{P}_q(\mathbf{x}_q(k_0+1), \mathsf{C}_q(Y_{\{-q\}}^*(k_0+1)))$ which is thus proved feasible. Then since $Y_q^*(k_0+1) \in \mathsf{C}_q(Y_{\{-q\}}^*(k_0+1))$ is directly enforced by constraint (12g) then the equivalence (18) proves (31). (32) follows from Proposition 3. □

Proposition 7 (Case 4: subsystem leaving) *If $p_{k_0} = q \neq 0$ and subsystem q leaves $\mathcal{P}_C(k_0+1) = \mathcal{P}_C(k_0)\setminus\{q\}$, then*

$$\{Y_p^*(k_0+1)\}_{p\in\mathcal{P}_C(k_0+1)} \in \bar{\mathsf{C}}_{n_C(k_0+1)} \tag{33}$$

and

$$Y_p^*(k_0+1) \in \{\mathbf{0}\}^{N+1}\ \forall p \in \mathcal{P}(k_0+1)\setminus\mathcal{P}_C(k_0+1) \tag{34}$$

Proof: Definition 7 ensures that any subset of feasible outputs will be also mutually feasible. Hence, $\{Y_p^*(k_0)\}_{p\in\mathcal{P}_C(k_0)\setminus q} \in \bar{\mathsf{C}}_{n_C(k_0)-1}$. Then from Proposition 4 it follows that adoption of the tails by all subsystems $p \in \mathcal{P}_C(k_0)\setminus q$ gives $\{Y_p^*(k_0+1)\}_{p\in\mathcal{P}_C(k_0)\setminus q} \in \bar{\mathsf{C}}_{n_C(k_0)-1}$ which is equivalent to (33). According to Step 22 of Algorithm 2, subsystem q can only leave if $Y_q^*(k_0+1) \in \{\mathbf{0}\}^{N+1}$ and $Y_p^*(k_0+1) \in \{\mathbf{0}\}^{N+1}\ \forall p \in \mathcal{P}(k_0)\setminus\mathcal{P}_c(k_0)$ follows from Proposition 3 hence (34) is satisfied since $\mathcal{P}(k_0+1)\setminus\mathcal{P}_C(k_0+1) = (\mathcal{P}(k_0)\setminus\mathcal{P}_c(k_0))\cup q$. □

Theorem 1 *Algorithm 2 ensures*

$$\{Y_p^*(k)\}_{p\in\mathcal{P}(k)} \in \bar{\mathsf{C}}_{n(k)} \tag{35}$$

and hence satisfaction of coupling constraints (2) for all $k > k_0$.

Proof: Propositions 4–7 have shown that conditions (33) and (34) are satisfied by recursion in every case, and Proposition 2 has shown that these conditions imply (35). □

6 Numerical Example

In this section, the proposed scalable DMPC scheme is applied to the air traffic problem in one airspace sector denoted by $\mathcal{A}$. Each aircraft has its own objective and dynamics. The coupling constraint has been expressed in Example 2 and set $\{\mathbf{0}\}$ has been defined in Example 4. Constraint satisfaction inside the controlled area is assured providing no two aircraft re-plan at the same time. The challenge is to handle the dynamic nature of the controlled area, agreeing a sequence for re-planning with aircraft constantly entering and leaving; this is done by self-organized sequencing.

ATM problem has been simulated using Dubins' car model with speed and turn curvature as control inputs, each subject to limits [Balluchi et al., 1996]. All aircraft have the constant altitude flight and the objective of each is to minimize the flight time from the initial point to the destination, although the non-linear optimizer will admit a wider variety of costs. All aircraft need to fly through the controlled area in order to reach the destination. For each aircraft, a collision free loitering circle found at the end of horizon defines its set of invariant states [Schouwenaars et al., 2004]. To provide differentiable representation of the avoidance constraints which is compatible with gradient-based nonlinear optimizer, exclusion regions are modeled by the polar set

method [Patel and Goulart, 2011].
The program has been executed for 1000sec and 80 aircraft have been added to the network in random times. The time horizon and the frame length both are 10 ($N = 10$, $L_f = 10$). Figure 1 shows a snapshot of the controlled area during running the program in which agent 2 is leaving the cooperation and agents 9, 10 and 11 are still in the safe mode and trying to get a time slot in the network for entering the cooperation. Figure 2 depicts generated trajectories of all aircraft. The relative distance between all pairs of agents during their flights in the controlled area is demonstrated in Figure 3. Although the pairwise separations in this figure are hard to observe in detail, the clear conclusion is that no trace ever goes below safety distance, verifying that separation has not lost between any pairs.

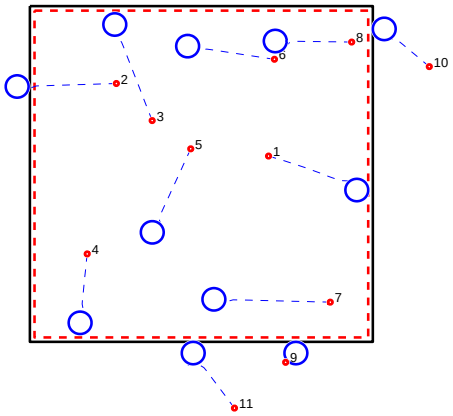

Figure 1. One snapshot from the controlled area

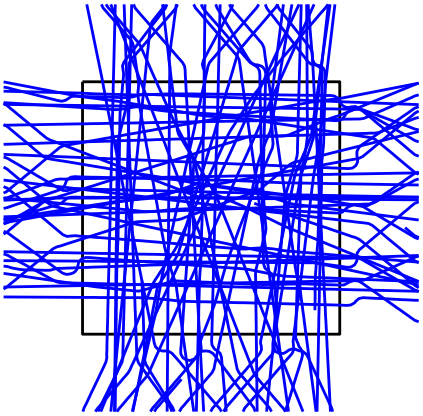

Figure 2. Generated trajectories by scalable DMPC

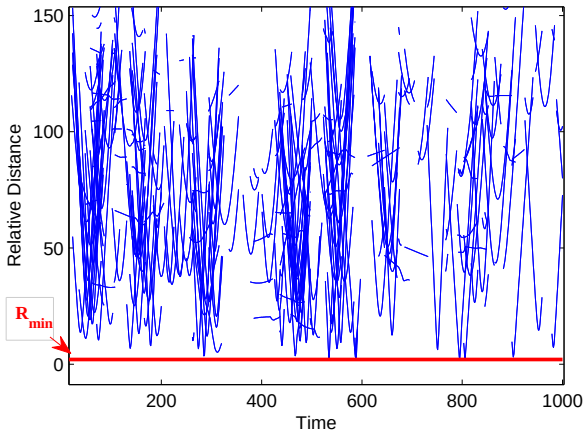

Figure 3. Relative distance between agents in the controlled area

7 Conclusion

In this work a distributed control algorithm has been presented for a system in which different subsystems join or leave dynamically. The coupling sources between the subsystems are output constraints which do not permit simultaneous re-planning. Serial optimization can solve this problem but would require a centralized coordination of the re-planning sequence which is not scalable. Instead, this work suggests that each subsystem finds its optimization slots in the network by following an algorithm based on the STDMA communication protocol. Although there is a certain level of conservativeness in any method working based on the serial scheme, scalable DMPC method gives the opportunity to do more with the limited frame length because subsystems can enter and leave. However, if subsystems try to enter the network faster than subsystems try to leave, a logged jam can happen and no further subsystem can enter.
Future work will explore the case of having several cooperation groups. This is motivated by the air traffic control problem, where aircraft in each airspace sector are considered as one cooperation group and aircraft can travel between different airspace sectors. Also, the possibility of having parallel optimizations for subsystems in different cooperation groups will be examined. Furthermore, sensitivity to the tuning parameters such as planning horizon, the frame length and rate of re-planning will be investigated.

References

Alessio, A., Barcelli, D., and Bemporad, A. (2011). Decentralized model predictive control of dynamically coupled linear systems. *Journal of Process Control*, 21(5):705–714.

Asadi, F. and Richards, A. (2015). Ad hoc distributed model predictive control of air traffic management. In

16th {IFAC} Workshop on Control Applications of Optimization CAO, volume 48, pages 68 – 73, Garmisch-Partenkirchen, Germany.

Balluchi, A., Bicchi, A., Balestrino, A., and Casalino, G. (1996). Path tracking control for dubin's cars. In *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, volume 4, pages 3123–3128. IEEE.

Barreiro-Gomez, J., Obando, G., Ocampo-Martinez, C., and Quijano, N. (2015). Making Non-Centralized a Model Predictive Control Scheme by Using Distributed Smith Dynamics. *IFAC-PapersOnLine*, 48(23):501–506.

Borrelli, F. and Keviczky, T. (2006). Distributed lqr design for dynamically decoupled systems. In *Decision and Control, 2006 45th IEEE Conference on*, pages 5639–5644. IEEE.

Bourdais, R., Buisson, J., Dumur, D., Guéguen, H., and Moroşan, P. (2014). Distributed mpc under coupled constraints based on dantzig-wolfe decomposition. In *Distributed Model Predictive Control Made Easy*, pages 101–114. Springer.

Christofides, P. D., Scattolini, R., de la Peña, D. M., and Liu, J. (2013). Distributed model predictive control: A tutorial review and future research directions. *Computers & Chemical Engineering*, 51:21–41.

Dai, L., Xia, Y., Gao, Y., Kouvaritakis, B., and Cannon, M. (2015). Cooperative distributed stochastic mpc for systems with state estimation and coupled probabilistic constraints. *Automatica*, 61:89–96.

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

Dunbar, W. B. and Murray, R. M. (2006). Distributed receding horizon control for multi-vehicle formation stabilization. *Automatica*, 42(4):549–558.

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

Gaugel, T., Mittag, J., Hartenstein, H., Papanastasiou, S., and Strom, E. G. (2013). In-depth analysis and evaluation of self-organizing tdma. In *Vehicular Networking Conference (VNC), 2013 IEEE*, pages 79–86. IEEE.

Grüne, L. and Pannek, J. (2011). *Nonlinear model predictive control: theory and algorithms*. Springer.

Hernandez, B. and Trodden, P. (2016). Distributed model predictive control using a chain of tubes. *arXiv preprint arXiv:1603.02044*.

Keviczky, T., Borrelli, F., and Balas, G. J. (2004a). Hierarchical design of decentralized receding horizon controllers for decoupled systems. In *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, volume 2, pages 1592–1597. IEEE.

Keviczky, T., Borrelli, F., and Balas, G. J. (2004b). A study on decentralized receding horizon control for decoupled systems. In *American Control Conference, 2004. Proceedings of the 2004*, volume 6, pages 4921–4926. IEEE.

Keviczky, T., Borrelli, F., and Balas, G. J. (2006). Decentralized receding horizon control for large scale dynamically decoupled systems. *Automatica*, 42(12):2105–2115.

Kuwata, Y., Richards, A., Schouwenaars, T., and How, J. P. (2007a). Distributed robust receding horizon control for multivehicle guidance. *Control Systems Technology, IEEE Transactions on*, 15(4):627–641.

Kuwata, Y., Richards, A., Schouwenaars, T., and How, J. P. (2007b). Distributed robust receding horizon control for multivehicle guidance. *Control Systems Technology, IEEE Transactions on*, 15(4):627–641.

Li, H., Shi, Y., and Yan, W. (2016). Distributed receding horizon control of constrained nonlinear vehicle formations with guaranteed γ-gain stability. *Automatica*, 68:148–154.

Lucia, S., Kögel, M., and Findeisen, R. (2015). Contract-based predictive control of distributed systems with plug and play capabilities. *IFAC-PapersOnLine*, 48(23):205–211.

Maciejowski, J. M. (2002). *Predictive control: with constraints*. Pearson education.

Mayne, D. Q., Rawlings, J. B., Rao, C. V., and Scokaert, P. O. (2000). Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814.

Müller, M. A., Reble, M., and Allgöwer, F. (2012). Cooperative control of dynamically decoupled systems via distributed model predictive control. *International Journal of Robust and Nonlinear Control*, 22(12):1376–1397.

Negenborn, R. R. and Maestre, J. (2014). Distributed model predictive control: An overview and roadmap of future research opportunities. *Control Systems, IEEE*, 34(4):87–97.

Patel, R. B. and Goulart, P. J. (2011). Trajectory generation for aircraft avoidance maneuvers using online optimization. *Journal of guidance, control, and dynamics*, 34(1):218–230.

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

Riverso, S., Farina, M., and Ferrari-Trecate, G. (2014). Plug-and-play model predictive control based on robust control invariant sets. *Automatica*, 50(8):2179–2186.

Rom, R. and Sidi, M. (2012). *Multiple access protocols: performance and analysis*. Springer Science & Business Media.

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

Schouwenaars, T., How, J., and Feron, E. (2004). Receding horizon path planning with implicit safety guarantees. In *American Control Conference, 2004. Proceedings of the 2004*, volume 6, pages 5576–5581. IEEE.

Stoustrup, J. (2009). Plug & Play Control: Control Technology Towards New Challenges. *European Journal*

of Control, 15(3-4):311–330.

Tedesco, F., Raimondo, D. M., and Casavola, A. (2014). Collision avoidance command governor for multi-vehicle unmanned systems. *International Journal of Robust and Nonlinear Control*, 24(16):2309–2330.

Trodden, P. and Richards, A. (2010). Distributed model predictive control of linear systems with persistent disturbances. *International Journal of Control*, 83(8):1653–1663.

Trodden, P. and Richards, A. (2013). Cooperative distributed mpc of linear systems with coupled constraints. *Automatica*, 49(2):479–487.

Wang, C. and Ong, C.-J. (2010). Distributed model predictive control of dynamically decoupled systems with coupled cost. *Automatica*, 46(12):2053–2058.

Zeilinger, M., Pu, Y., Riverso, S., Ferrari-Trecate, G., and Jones, C. (2013). Plug and play distributed model predictive control based on distributed invariance and optimization. *52nd IEEE Conference on Decision and Control*, pages 5770–5776.