TU Delft

Delft University of Technology

Distributed Model Predictive Control for Piecewise Affine Systems Based on Switching ADMM

Mallick, Samuel; Dabiri, Azita; De Schutter, Bart

DOI
10.1109/TAC.2024.3512334

Publication date
2025

Document Version
Final published version

Published in
IEEE Transactions on Automatic Control

Citation (APA)
Mallick, S., Dabiri, A., & De Schutter, B. (2025). Distributed Model Predictive Control for Piecewise Affine Systems Based on Switching ADMM. *IEEE Transactions on Automatic Control*, *70*(6), 3727-3741. https://doi.org/10.1109/TAC.2024.3512334

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.
For technical reasons the number of authors shown on this cover page is limited to a maximum of 10.

Distributed Model Predictive Control for Piecewise Affine Systems Based on Switching ADMM

Samuel Mallick, Azita Dabiri, and Bart De Schutter, *Fellow, IEEE*

***Abstract*—In this article, we present a novel approach for distributed model predictive control (MPC) for piecewise affine (PWA) systems. Existing approaches rely on solving mixed-integer optimization problems, requiring significant computation power or time. We propose a distributed MPC scheme that requires solving only convex optimization problems. The key contribution is a novel method, based on the alternating direction method of multipliers, for solving the nonconvex optimal control problem that arises due to the PWA dynamics. We present a distributed MPC scheme, leveraging this method, that explicitly accounts for the coupling between subsystems by reaching agreement on the values of coupled states. Stability and recursive feasibility are shown under additional assumptions on the underlying system. Two numerical examples are provided, in which the proposed controller is shown to significantly improve the CPU time and closed-loop performance over existing state-of-the-art approaches.**

***Index Terms*—Alternating direction method of multipliers (ADMM), distributed model predictive control (MPC), networked systems, piecewise affine (PWA) systems.**

I. Introduction

DISTRIBUTED control of large-scale hybrid systems is an important challenge in the field of systems and control [1]. Hybrid systems are dynamical systems exhibiting both continuous and discrete dynamics. They are powerful models for many real-world large-scale systems, such as transportation networks [2], power networks [3], and water networks [4]. In particular, piecewise affine (PWA) models are a popular class of hybrid system model due their suitability for analysis [5], proven equivalence to other hybrid model classes [6], and ability to approximate nonlinear functions to arbitrary accuracy.

Distributed model predictive control (MPC) [7] is a promising candidate for control of large-scale PWA systems. MPC is an optimization-based paradigm where control inputs are generated online by minimizing a sum of stage costs over a prediction horizon, subject to the system dynamics, and physical and operational constraints [8]. Distributed MPC applies this idea to spatially separated or decomposed systems, where subsystem controllers solve, sometimes iteratively, local optimization problems, and where the coupling between connected subsystems is accounted for by communication [7]. For linear systems, this approach can reconstruct the solution to a globally defined convex MPC controller through the use of iterative distributed optimization algorithms, e.g., the alternating direction method of multipliers (ADMM) [9]. In this case, as the globally defined controller considers the global system, the effect of coupling between subsystems is accounted for and the stability and performance of the globally defined MPC controller carries through to the distributed implementation. For PWA systems, however, the PWA dynamics result in a nonconvex global optimization problem for which distributed optimization algorithms, such as ADMM, are not guaranteed to recover the global optimum, nor even to find a feasible solution. As such, the effects of coupling between subsystems may cause instability or negatively affect the performance of the resulting distributed controller. Furthermore, the local optimization problems are also nonconvex, and the computational burden of solving them iteratively can become prohibitive, particularly as their computational complexity grows exponentially with the size of the prediction horizon [10]. Therefore, two key challenges for distributed MPC for PWA systems are: how to reduce the online computational burden, and how to account for the effects of coupling between subsystems.

The few existing approaches for distributed MPC for PWA systems rely on converting the PWA dynamics into mixed logical dynamical (MLD) form, converting the MPC optimization problems into mixed-integer quadratic problems (MIQPs) [10]. Distributed control is then conducted by the subsystems solving local MIQPs, iteratively in parallel [11], or in a sequential order [12], with communication of the local solutions between subsystems handling the coupling. However, these approaches suffer from the computational complexity of solving multiple MIQPs. To reduce the online computational burden, [3] proposes to select values for the integers in the MIQPs with heuristics, using convex distributed MPC techniques to then solve for the values of the continuous variables. However, the heuristic choice of integer solutions is in general highly suboptimal. Perhaps the most theoretically complete approach was proposed recently in [13], where distributed MPC for PWA systems is

Received 26 September 2024; accepted 30 November 2024. Date of publication 5 December 2024; date of current version 30 May 2025. This work was supported by the European Research Council (ERC) through the European Union's Horizon 2020 research and innovation programme under Grant 101018826-CLariNet. Recommended by Associate Editor Sergey N. Dashkovskiy. *(Corresponding author: Samuel Mallick.)*

The authors are with the Delft Center for Systems and Control, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: s.h.mallick@tudelft.nl; a.dabiri@tudelft.nl; b.deschutter@tudelft.nl).

Digital Object Identifier 10.1109/TAC.2024.3512334

© 2024 The Authors. This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see http://creativecommons.org/licenses/by/4.0/

presented using robustly controllable sets. The computational complexity of the local MIQPs is reduced by considering a prediction horizon of length one, and convergence to a terminal set is proven. However, the coupling between subsystems is handled using robustness techniques, where subsystems consider coupling effects as disturbances, rather than agreeing on the values of the shared states. This adds conservativeness, reducing performance, and limiting the approach to systems with weak coupling.

In light of these challenges, this article presents a novel approach for distributed MPC for PWA systems that systematically handles the coupling between subsystems and relies only on convex optimization. The key contribution, enabling the approach, is a novel distributed method for solving the global nonconvex optimization problem arising in the globally formulated MPC controller. This method is based on ADMM, and leverages the underlying PWA dynamics of the system to solve the nonconvex optimization problem distributively using only convex optimization. Furthermore, the coupling between subsystems is handled explicitly, with the distributed solutions provably converging to points where the values of all shared states are agreed upon. As such, the proposed method can improve upon the performance of existing controllers that heuristically handle coupling between subsystems, and presents a significantly lower computational burden than approaches based on mixed-integer optimization.

The rest of this article is organized as follows. Section II gives the preliminaries, formalizing the systems considered and giving background on centralized and distributed MPC. Section III presents the switching ADMM procedure and the distributed MPC controller, with the stabilizing properties proved in Section IV. In Section V, two numerical examples demonstrate the effectiveness of the approach. Finally, Section VI concludes this article.

II. Preliminaries

A. Notation

Define the following sets, all subsets of $\mathbb{N}$, as $\mathcal{M} = \{1, \dots, M\}, \mathcal{L}_i = \{1, \dots, L_i\}, \mathcal{K} = \{0, \dots, N-1\}$, and $\mathcal{K}' = \{0, \dots, N\}$. We use the counter t to represent closed-loop time steps and k for time steps within an MPC prediction horizon. A vector that stacks the vectors x_i, $i \in \mathcal{M}$, in one column vector is denoted $\text{col}_{i\in\mathcal{M}}(x_i)$. Define the closure of set $\mathcal{P}$ as $\overline{\mathcal{P}}$. For brevity we write $(A)^\top B(A)$ as $(\star)^\top B(A)$.

B. Distributed PWA Systems

We consider a global system composed of M subsystems where each subsystem $i \in \mathcal{M}$ has a state $x_i \in \mathbb{R}^{n_i}$ and an input $u_i \in \mathbb{R}^{m_i}$. Both states and inputs are subject to convex constraints

$$x_i \in \mathcal{X}_i, \quad u_i \in \mathcal{U}_i \quad \forall i \in \mathcal{M}. \tag{1}$$

The graph $\mathcal{G} = (\mathcal{M}, \mathcal{E})$ defines a coupling topology where the edges $\mathcal{E}$ are ordered pairs (i, j) indicating that subsystem i effects subsystem j through the dynamics, stage cost, constraints, or some combination thereof. Define the neighborhood of subsystem i as $\mathcal{N}_i = \{j \in M | (j, i) \in \mathcal{E}, i \neq j\}$. A subsystem is not in its own neighborhood, i.e., $(i, i) \notin \mathcal{E}$. It is assumed that subsystem i and subsystem j can communicate in a bidirectional manner if $i \in \mathcal{N}_j$ or $j \in \mathcal{N}_i$.

The subsystems are governed by discrete-time PWA dynamics,[1] with the state update equations f_i affine on a finite number of convex polytopes, $\{P_i^{(l)}\}_{l\in\mathcal{L}_i}$, referred to as regions

$$\begin{aligned} x_i^+ &= f_i\left(x_i, u_i, \{x_j\}_{j\in\mathcal{N}_i}\right) \\ &= A_i^{(l)} x_i + B_i^{(l)} u_i + c_i^{(l)} \\ &\quad + \sum_{j\in\mathcal{N}_i} A_{ij}^{(l)} x_j, \; x_i \in P_i^{(l)} \quad \forall l \in \mathcal{L}_i. \end{aligned} \tag{2}$$

The regions have nonoverlapping interiors, i.e., $\text{int}(P_i^{(l)}) \cap \text{int}(P_i^{(l')}) = \emptyset$ for $l, l' \in \mathcal{L}_i, l \neq l'$, and form a partition of the state space, i.e., $\bigcup_{l\in\mathcal{L}_i} P_i^{(l)} = \mathcal{X}_i$. Define the global state and input, the aggregation of states and inputs for all subsystems, as $x = \text{col}_{i\in\mathcal{M}}(x_i)$ and $u = \text{col}_{i\in\mathcal{M}}(u_i)$. Furthermore, define variables that gather states and inputs over the prediction horizon: $\mathbf{x}_i = (x_i^\top(0), \dots, x_i^\top(N))^\top$, $\mathbf{u}_i = (u_i^\top(0), \dots, u_i^\top(N-1))^\top$, $\mathbf{x} = (x^\top(0), \dots, x^\top(N))^\top$, and $\mathbf{u} = (u^\top(0), \dots, u^\top(N-1))^\top$.

C. Centralized MPC

Consider the global MPC controller defined by the following finite-horizon optimal control problem parameterized by x

$$\mathcal{P}(x): \min_{\mathbf{x},\mathbf{u}} \sum_{i\in\mathcal{M}} F_i\left(\mathbf{x}_i, \mathbf{u}_i, \{\mathbf{x}_j\}_{j\in\mathcal{N}_i}\right) \tag{3a}$$

s.t. $\forall i \in \mathcal{M}$:

$$x_i(k+1) = f_i\left(x_i(k), u_i(k), \{x_j(k)\}_{j\in\mathcal{N}_i}\right) \quad \forall k \in \mathcal{K} \tag{3b}$$

$$h_i\left(x_i(k), \{x_j(k)\}_{j\in\mathcal{N}_i}\right) \leq 0 \quad \forall k \in \mathcal{K}' \tag{3c}$$

$$(x_i(k), u_i(k)) \in \mathcal{X}_i \times \mathcal{U}_i \quad \forall k \in \mathcal{K}' \tag{3d}$$

$$x_i(0) = x_i \tag{3e}$$

with

$$\begin{aligned} &F_i\left(\mathbf{x}_i, \mathbf{u}_i, \{\mathbf{x}_j\}_{j\in\mathcal{N}_i}\right) = \\ &\sum_{k\in\mathcal{K}} \ell_i\left(x_i(k), u_i(k), \{x_j(k)\}_{j\in\mathcal{N}_i}\right) + V_{\text{f},i}\left(x_i(N)\right). \end{aligned} \tag{4}$$

The convex functions ℓ_i and $V_{\text{f},i}$ are stage and terminal costs, and the linear functions h_i define coupled convex inequality constraints. The first entries in the optimal input sequences define feedback control laws $u_{\text{MPC},i}(x) = u_i^\star(0)$.

Define the set of states x for which $\mathcal{P}$ is feasible as $\mathcal{X}_0$. For a global state $x \in \mathcal{X}_0$ and control sequence $\mathbf{u}$ define the value as

$$V(x, \mathbf{u}) = \sum_{i\in\mathcal{M}} F_i\left(\mathbf{x}_i, \mathbf{u}_i, \{\mathbf{x}_j\}_{j\in\mathcal{N}_i}\right) \tag{5}$$

[1] As only discrete-time systems are considered, hybrid phenomena such as arbitrarily fast switching and Zeno behavior are not relevant [14].

if, for all $i \in \mathcal{M}$ and $k \in \mathcal{K}'$, $h_i(x_i(k), \{x_j(k)\}_{j\in\mathcal{N}_i}) \leq 0$ and $(x_i(k), u_i(k)) \in \mathcal{X}_i \times \mathcal{U}_i$, and $V(x, \mathbf{u}) = \infty$ otherwise, with the state sequences $\mathbf{x}_i$ generated by applying $\mathbf{u}$ to the dynamics (2) starting from x. Define a globally feasible set of local control sequences as follows.

Definition 1 (Globally feasible local control sequences): The set of local control sequences $\{\mathbf{u}_i\}_{i\in\mathcal{M}}$ is *globally feasible* for x if the corresponding global control sequence $\mathbf{u}$ is feasible for $\mathcal{P}(x)$, i.e., $V(x, \mathbf{u}) < \infty$.

The MPC scheme defined by $\mathcal{P}$ considers general coupling between subsystems, i.e., in the costs, dynamics, and constraints. Solving $\mathcal{P}$ directly is undesirable as the complexity of the optimization problem grows with the number of subsystems. In addition, a centralized solution requires centralized coordination, which cannot preserve privacy of the local functions of subsystems, and presents a single point of failure. It is therefore desirable to solve $\mathcal{P}$ distributively, where each subsystem computes a local control input using only local information and communication with its neighbors.

Remark 1: The approach presented in this article extends easily to the case where the cost functions F_i also switch depending on the subsystem's PWA region. For clarity of exposition we do not explicitly include this case in $\mathcal{P}$, as the notation would then become unwieldy.

D. Distributed MPC

By introducing copies of coupled states the problem $\mathcal{P}$ can be reformulated for distributed optimization [15], [16], [17]. Define an augmented state trajectory for each subsystem that includes copies of the state trajectories of neighboring subsystems

$$\tilde{\mathbf{x}}_i = \left(\mathbf{x}_i^\top, \mathrm{col}^\top_{j\in\mathcal{N}_i}(\mathbf{x}_j^{(i)})\right)^\top \tag{6}$$

where $\mathbf{x}_j^{(i)}$ is agent i's local copy of agent j's state trajectory. Furthermore, define a global augmented state as $\tilde{\mathbf{x}} = \mathrm{col}_{i\in\mathcal{M}}(\tilde{\mathbf{x}}_i)$. An equivalent problem to $\mathcal{P}$ is then

$$\mathcal{P}_\mathrm{d}(x): \min_{\tilde{\mathbf{x}},\mathbf{u}} \sum_{i\in\mathcal{M}} F_i\left(\mathbf{x}_i, \mathbf{u}_i, \{\mathbf{x}_j^{(i)}\}_{j\in\mathcal{N}_i}\right) \tag{7a}$$

s.t. $\forall i \in \mathcal{M}$:

$$x_i(k+1) = f_i\left(x_i(k), u_i(k), \{x_j^{(i)}(k)\}_{j\in\mathcal{N}_i}\right) \quad \forall k \in \mathcal{K} \tag{7b}$$

$$h_i\left(x_i(k), \{x_j^{(i)}(k)\}_{j\in\mathcal{N}_i}\right) \leq 0 \quad \forall k \in \mathcal{K}' \tag{7c}$$

(3d), (3e) (7d)

$$x_j^{(i)}(k) = x_j(k),\ j \in \mathcal{N}_i \quad \forall k \in \mathcal{K}'. \tag{7e}$$

The equality constraint (7e) gives consistency between the true state trajectories and their copies and therefore equivalence between $\mathcal{P}$ and $\mathcal{P}_\mathrm{d}$. In $\mathcal{P}_\mathrm{d}$ the stage costs, dynamics, and inequality constraints can be separated across the local variables $\tilde{\mathbf{x}}_i$ and $\mathbf{u}_i$. The ADMM [9] can then be applied to solve $\mathcal{P}_\mathrm{d}$ distributively by dualizing the consistency constraint (7e), and having systems iteratively solve and communicate the solutions to local subproblems derived from $\mathcal{P}_\mathrm{d}$. However, due to the PWA dynamics (7b) $\mathcal{P}_\mathrm{d}$ is nonconvex, and ADMM is hence not guaranteed to converge to an optimal, or even feasible, solution. In addition, systems must solve nonconvex local optimization problems iteratively, requiring significant computation time or power. To address this, in the following section we introduce distributed MPC based on a switching ADMM procedure that, leveraging the structure of the PWA dynamics, can guarantee convergence to a feasible solution, and requires solving only convex optimization problems.

III. Switching ADMM-Based Distributed MPC

In this section, we introduce the proposed distributed MPC algorithm. First, we show that the structure of $\mathcal{P}_\mathrm{d}$ is piecewise-convex over a finite collection of polytopes. This structure is then leveraged to formulate the switching ADMM procedure upon which the distributed MPC algorithm is based.

A. Structure of $\mathcal{P}_d$

The following analysis is inspired by the *reverse transformation* presented in [18], where MPC for a single PWA system was considered. Define concatenated decision variables as $\mathbf{y}_i = (\tilde{\mathbf{x}}_i^\top, \mathbf{u}_i^\top)^\top$ and $\mathbf{y} = (\tilde{\mathbf{x}}^\top, \mathbf{u}^\top)^\top$. The feasible set for $\mathcal{P}_\mathrm{d}(x)$ is then

$$\mathcal{Y}(x) = \left\{\mathbf{y} = (\tilde{\mathbf{x}}^\top, \mathbf{u}^\top)^\top \middle| \text{(7b)–(7e)} \quad \forall i \in \mathcal{M}\right\}. \tag{8}$$

We now introduce the notion of *switching sequences* that specify a sequence of PWA regions over the prediction horizon. Consider the switching sequence $s_i \in \mathcal{S}_i = \{1, \ldots, L_i\}^{N+1}$ that specifies the PWA regions for subsystem i as

$$\begin{aligned} &s_i = s_i(0), s_i(1), \ldots, s_i(N) \\ &\Rightarrow x_i(k) \in P_i^{(s_i(k))} \quad \forall k \in \mathcal{K}' \end{aligned} \tag{9}$$

and the corresponding time-varying linear dynamics

$$\begin{aligned} x_i(k+1) &= f_i^{(s_i)}\left(x_i(k), u_i(k), \{x_j^{(i)}(k)\}_{j\in\mathcal{N}_i}\right) \\ &= A_i^{(s_i(k))}x_i(k) + B_i^{(s_i(k))}u_i(k) \\ &\quad + c_i^{(s_i(k))} + \sum_{j\in\mathcal{N}_i} A_{ij}^{(s_i(k))}x_j^{(i)}(k) \quad \forall k \in \mathcal{K}. \end{aligned} \tag{10}$$

A particular $\mathbf{x}_i$ is said to *generate* a switching sequence s_i if

$$x_i(k) \in \overline{P}_i^{(s_i(k))} \quad \forall k \in \mathcal{K}'. \tag{11}$$

Note that more than one s_i can be generated by $\mathbf{x}_i$ if $x_i(k)$ lies on the boundary of at least two PWA regions for some k, i.e., if $x_i(k) \in \overline{P}_i^{(a)} \cap \overline{P}_i^{(b)}$ then two sequences are generated, one containing $s_i(k) = a$ and the other $s_i(k) = b$. For a given x_i, $\mathbf{u}_i$, and $\{\mathbf{x}_j^{(i)}\}_{j\in\mathcal{N}_i}$ subsystem i can evaluate the generated switching sequences by rolling out and branching its dynamics as in Algorithm 1.

Let $s = (s_1, \ldots, s_M) \in \mathcal{S}$, $\mathcal{S} = \mathcal{S}_1 \times \cdots \times \mathcal{S}_M$, define a global switching sequence specifying local switching sequences for each subsystem. This global s defines a new optimal control

Algorithm 1: Eval-Switching.

Inputs: State x_i, input sequence $\mathbf{u}_i$, and coupled state copies $\{\mathbf{x}_j^{(i)})\}_{j\in\mathcal{N}_i}$
Initialize: $\Phi \leftarrow \{(s_i = (1,\dots,1), x_i)\}$, set of tuples with sequences (1's as dummy variables) and states
for $k = 0,\dots,N$ **do**
  $\Phi_{\text{branch}} \leftarrow \{\}$
  for $\phi = (s_i, x_i) \in \Phi$ **do**
    first-flag $\leftarrow 1$
    for $l \in \mathcal{L}_i$ s.t. $x_i \in \overline{P}_i^{(l)}$ **do**
      if first-flag $= 1$ **then**
        first-flag $\leftarrow 0$
        $s_i(k) \leftarrow l$
        if $k < N$ **then**
          $x_i \leftarrow f_i^{(l)}(x_i, u_i(k), \{x_j^{(i)}(k)\}_{j\in\mathcal{N}_i})$
        end if
      else
        $(s_i', x_i') \leftarrow \phi$
        $s_i'(k) \leftarrow l$
        if $k < N$ **then**
          $x_i' \leftarrow f_i^{(l)}(x_i', u_i(k), \{x_j^{(i)}(k)\}_{j\in\mathcal{N}_i})$
        end if
        $\Phi_{\text{branch}} \leftarrow \Phi_{\text{branch}} \cup \{(s_i', x_i')\}$
      end if
    end for
  end for
  $\Phi \leftarrow \Phi \cup \Phi_{\text{branch}}$
end for
Outputs: $\{s_i\}_{\phi\in\Phi}$

problem

$$\mathcal{P}_s(x): \min_{\mathbf{y}} \sum_{i\in\mathcal{M}} F_i\left(\mathbf{x}_i, \mathbf{u}_i, \{\mathbf{x}_j^{(i)}\}_{j\in\mathcal{N}_i}\right) \tag{12a}$$

$$\text{s.t.} \quad \forall i \in \mathcal{M}:$$

$$\text{(7c)}, -\text{(7e)}, \text{(10)} \tag{12b}$$

$$x_i(k) \in P_i^{(s_i(k))} \quad \forall k \in \mathcal{K}' \tag{12c}$$

where the time-varying dynamics (10) replace the PWA dynamics and the PWA regions selected by s are enforced with the additional constraints (12c). The feasible set for $\mathcal{P}_s(x)$ is

$$\mathcal{Y}_s(x) = \left\{\mathbf{y} = \tilde{\mathbf{x}}^\top, \mathbf{u}^\top)^\top \middle| \text{(7c)} - -\text{(7e)}, \text{(10)}, \text{(12c)} \quad \forall i \in \mathcal{M}\right\} \tag{13}$$

and it is convex as the constraints (7c)–(7e) are convex, the dynamics (10) are linear, and the PWA regions in (12c) are convex. Note that $\mathcal{Y}_s$ is empty for many choices of s. We now show that $\mathcal{Y}$ is the union of the sets $\mathcal{Y}_s$.

Lemma 1: Suppose x and s are given such that $\mathcal{Y}_s(x) \neq \emptyset$. Then, for all $\mathbf{y} = (\tilde{\mathbf{x}}^\top, \mathbf{u}^\top)^\top \in \mathcal{Y}_s(x)$

$$f_i\left(x_i(k), u_i(k), \{x_j^{(i)}(k)\}_{j\in\mathcal{N}_i}\right) = f_{i,s}\left(x_i(k), u_i(k), \{x_j^{(i)}(k)\}_{j\in\mathcal{N}_i}\right) \tag{14}$$

$\forall k \in \mathcal{K} \forall i \in \mathcal{M}$.

Proof: Let x and s be given, and let $\mathbf{y}$ be an arbitrary element of $\mathcal{Y}_s(x)$. From (12c), $x_i(k) \in P_i^{(s_i(k))}\ \forall k \in \mathcal{K} \forall i \in \mathcal{M}$, so that, from (2)

$$\begin{aligned} &f_i\left(x_i(k), u_i(k), \{x_j^{(i)}(k)\}_{j\in\mathcal{N}_i}\right) \\ &= \left(A_i^{(l)}x_i + B_i^{(l)}u_i + \sum_{j\in\mathcal{N}_i} A_{ij}^{(l)}\right)\Big|_{l=s_i(k)} \\ &= f_{i,s}\left(x_i(k), u_i(k), \{x_j^{(i)}(k)\}_{j\in\mathcal{N}_i}\right) \end{aligned} \tag{15}$$

for all $k \in \mathcal{K}$ and $i \in \mathcal{M}$. □

Proposition 1: The feasible set for problem $\mathcal{P}_\text{d}(x)$ is the union of a finite number of convex sets, that are the feasible sets for the problems $\mathcal{P}_s(x)$, i.e.,

$$\mathcal{Y}(x) = \bigcup_{s\in\mathcal{S}} \mathcal{Y}_s(x). \tag{16}$$

Proof: 1) Suppose $\mathbf{y} \in \mathcal{Y}(x)$. Let s be a switching sequence generated by $\mathbf{y}$, so that $x_i(k) \in P_i^{(s_i(k))}\ \forall k \in \mathcal{K} \forall i \in \mathcal{M}$. Hence, $\mathbf{y}$ satisfies (12c) in $\mathcal{P}_s(x)$. Furthermore, the dynamics in (2) reduce to (10). Finally, $\mathbf{y}$ satisfies (7c)–(7e) by definition. Thus, $\mathbf{y} \in \mathcal{Y}_s(x)$ and $\mathcal{Y}(x) \subseteq \mathcal{Y}_s(x) \subseteq \bigcup_{s\in\mathcal{S}} \mathcal{Y}_s(x)$. 2) Now suppose $\mathbf{y} \in \bigcup_{s\in\mathcal{S}} \mathcal{Y}_s(x)$. Then, there exists $s \in \mathcal{S}^M$ such that $\mathbf{y} \in \mathcal{Y}_s(x)$. By Lemma 1, for all $\mathbf{y} \in \mathcal{Y}_s(x)$, (10) is equivalent to the dynamics in (2), and $\mathbf{y}$ satisfies (7b). In addition, $\mathbf{y}$ satisfies (7c)–(7e) by definition. Thus, $\mathbf{y} \in \mathcal{Y}(x)$ so that $\bigcup_{s\in\mathcal{S}} \mathcal{Y}_s(x) \subseteq \mathcal{Y}(x)$. (16) follows. □

Proposition 1 reveals the structure of $\mathcal{P}_\text{d}$. It is piecewise convex, with each convex region associated with a global switching sequence s and being defined over the set $\mathcal{Y}_s$. It follows that, in theory, $\mathcal{P}_\text{d}$ can be solved by solving $\mathcal{P}_s$ for each $s \in \mathcal{S}$, and taking the solution that gives the lowest cost. However, in practice, this is not feasible as $|S|$ grows exponentially with M and N.

B. Switching ADMM Procedure

Rather than solving every convex piece $\mathcal{P}_s$, we propose to consider one piece at a time. Applying ADMM to one $\mathcal{P}_s$ will converge to the optimum of that convex problem. However, during the iterations it can be beneficial to switch to a new $\mathcal{P}_s$ that gives a lower cost. The proposed idea is then to allow subsystems, during the ADMM iterations, to locally change switching sequences s_i, changing the global s and $\mathcal{P}_s$, thereby moving from one convex piece to another convex piece of $\mathcal{P}_\text{d}$, until a local minimum is found. In this section, we formalize this idea.

First, we make the ADMM procedure for solving $\mathcal{P}_s$ explicit. Define the local constraint set containing the constraints independent of s_i as

$$\mathcal{Y}_i(x_i) = \left\{\mathbf{y}_i \middle| \text{(7c)}, \text{(7d)}\right) \tag{17}$$

and the local constraint set containing the constraints dependent on s_i as

$$\mathcal{Y}_i^{(s_i)} = \left\{\mathbf{y}_i \middle| \text{(10)}, \text{(12c)}\right\}. \tag{18}$$

Introduce additional copies of each state trajectory $\mathbf{z} = (\mathbf{z}_1^\top, \mathbf{z}_2^\top, \ldots, \mathbf{z}_M^\top)^\top$, with the relevant components of $\mathbf{z}$ for subsystem i denoted $\tilde{\mathbf{z}}_i = (\mathbf{z}_i^\top, \mathrm{col}_{j\in\mathcal{N}_i}^\top(\mathbf{z}_j))^\top$. The problem $\mathcal{P}_s(x)$ can then be reformulated as

$$\begin{aligned}\min_{\{\mathbf{y}_i\}_{i\in\mathcal{M}}} \sum_{i\in\mathcal{M}} F_i\left(\mathbf{x}_i, \mathbf{u}_i, \{\mathbf{x}_j^{(i)}\}_{j\in\mathcal{N}_i}\right)\\ \text{s.t.} \quad \forall i \in \mathcal{M}:\\ \mathbf{y}_i \in \mathcal{Y}_i(x_i) \cup \mathcal{Y}_i^{(s_i)}\\ \tilde{\mathbf{x}}_i = \tilde{\mathbf{z}}_i \end{aligned} \tag{19}$$

with the final constraint forcing agreement across all coupled states. Recall that in (19) the switching sequences s_i are not optimization variables and have been prespecified.

ADMM solves (19) with the following distributed iterations [9]:

$$\begin{aligned}\mathbf{y}_i^{\tau+1} = \underset{\mathbf{y}_i\in\mathcal{Y}_i\cup\mathcal{Y}_i^{(s_i)}}{\arg\min} \; F_i\left(\mathbf{x}_i, \mathbf{u}_i, \{\mathbf{x}_j^{(i)}\}_{j\in\mathcal{N}_i}\right) + (\boldsymbol{\lambda}_i^\tau)^\top \tilde{\mathbf{x}}_i\\ + \frac{\rho}{2}\|\tilde{\mathbf{x}}_i - \tilde{\mathbf{z}}_i^\tau\|_2^2 \end{aligned} \tag{20a}$$

$$\mathbf{z}_i^{\tau+1} = \frac{1}{|\mathcal{N}_i|+1}\left(\mathbf{x}_i^{\tau+1} + \sum_{j\in\mathcal{N}_i} \mathbf{x}_i^{(j),\tau+1}\right) \tag{20b}$$

$$\boldsymbol{\lambda}_i^{\tau+1} = \boldsymbol{\lambda}_i^\tau + \rho(\tilde{\mathbf{x}}_i^{\tau+1} - \tilde{\mathbf{z}}_i^{\tau+1}) \tag{20c}$$

with $\boldsymbol{\lambda}_i$ the Lagrange multiplier for subsystem i and $\rho > 0$ a penalty parameter. This is a distributed message passing algorithm where, in step (20a) each subsystem solves a local optimization problem, in step (20b) the local solutions are communicated and averaged between neighboring subsystems, and in step (20c) the subsystems update their multipliers locally. Define the residual $r(\mathbf{y})$ as the summed disagreement on the values of shared states

$$r(\mathbf{y}) = \sum_{i\in\mathcal{M}}\sum_{j\in\mathcal{N}_i} \|\mathbf{x}_j^{(i)} - \mathbf{x}_j\|_2. \tag{21}$$

As $\tau \to \infty$ the local solutions converge to the minimizers of $\mathcal{P}_s$, i.e., $\mathbf{y}_i^\tau \to \mathbf{y}_i^\star, \forall i \in \mathcal{M}$, and the shared states converge to agreement [9]

$$\mathbf{x}_j^{(i),\tau} - \mathbf{x}_j^\tau \to 0 \quad \forall j \in \mathcal{N}_i \quad \forall i \in \mathcal{M} \tag{22}$$

i.e., $r(\mathbf{y}^\tau) \to 0$. In addition, as the optimizer of $\mathcal{P}_s$ is trivially a feasible point of $\mathcal{Y}_s$, and $\mathcal{Y}_s \subset \mathcal{Y}$, we have that as $\tau \to \infty$

$$\mathbf{y}^\tau \in \mathcal{Y}. \tag{23}$$

In the following, we will refer to any point where (22) and (23) hold as a *consensus point*.

Our key insight is that new local switching sequences can be identified when the local solutions to (20a) are pushed to the boundary of the feasible set $\mathcal{Y}_i^{(s_i)}$. Indeed, in the case where the current $\mathcal{P}_s$ does not contain a local minimum of $\mathcal{P}_\mathrm{d}$ this is the expected behavior, as the minimum of $\mathcal{P}_s$ is then on the boundary of its feasible set. When this occurs, subsystems change their local switching sequences s_i, which globally changes s and $\mathcal{P}_s$. In this way, the nonconvex problem $\mathcal{P}_\mathrm{d}$ is explored by jumping between the different approximations $\mathcal{P}_s$, with the jumping determined distributively by intermediate local solutions to (20a). Once local solutions are no longer pushed to boundaries of $\mathcal{Y}_i^{(s_i)}$ the system-wide $\mathcal{P}_s$ no longer changes, and the ADMM iterations will converge, as $\mathcal{P}_s$ is convex, to a local minimum of $\mathcal{P}_\mathrm{d}$ and a consensus point.

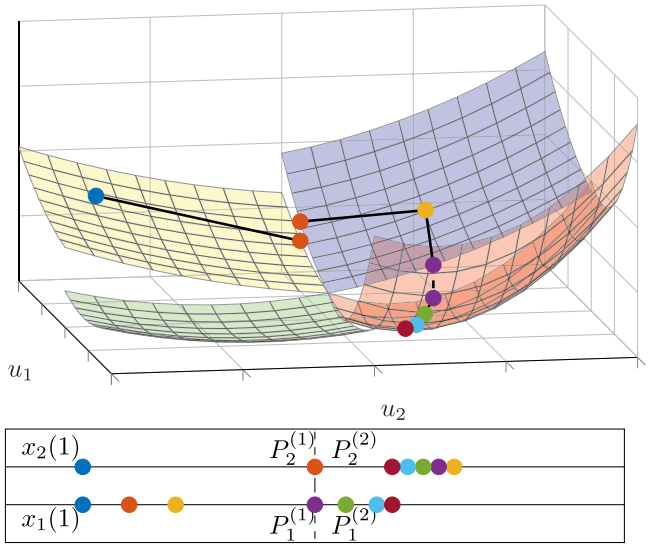

Fig. 1. Visualization of switching ADMM procedure.

Let us elaborate with the representative visual example in Fig. 1 that depicts two systems, each with two PWA regions $P_i^{(1)}$ and $P_i^{(2)}$ and current states $x_1 \in P_1^{(1)}$ and $x_2 \in P_2^{(1)}$. To facilitate visualization consider scalar states and inputs $u_i, x_i \in \mathbb{R}$ and a horizon of $N = 1$, such that the control input decision variables for each subsystem are scalar and the switching sequences are of length two. There are 16 possible global switching sequences

$$\mathcal{S} = \{s = (s_1, s_2) | s_1, s_2 \in \{(1,1),(1,2),(2,1),(2,2)\}\}. \tag{24}$$

However, as $x_1 \in P_1^{(1)}$ and $x_2 \in P_2^{(1)}$, only switching sequences with $s_1(0) = s_2(0) = 1$ correspond to a nonempty $\mathcal{P}_s$. The top plot of Fig. 1 shows, over the control input decision space, the four nonempty convex pieces, $\{\mathcal{P}_s\}_{s\in\hat{\mathcal{S}}}, \hat{\mathcal{S}} = \{(s_1, s_2)|s_1, s_2 \in \{(1,1),(1,2)\}\}$, constituting $\mathcal{P}_\mathrm{d}$. The bottom plot shows the local decision variables $x_1(1)$ and $x_2(1)$ and the PWA partitions. An initial guess $\hat{u}_1(0)$ and $\hat{u}_2(0)$ generates $x_1(1) \in P_1^{(1)}$ and $x_2(1) \in P_2^{(1)}$ (blue dot), and thus $s_1 = s_2 = (1,1)$. Each subsystem sets the constraint set $\mathcal{Y}_i^{(s_i)}$ that defines the problem $\mathcal{P}_s$ (yellow region). The subsystems commence the ADMM iterations performing one round of the steps in (20). The output of the local minimization yields new solutions (orange dot), and subsystem 2 identifies that this local solution is on the boundary of $\overline{P}_2^{(1)}$ and $\overline{P}_2^{(2)}$ using Algorithm 1. Subsystem 2 then changes its switching sequence to $s_2 = (1,2)$, changing the local constraint set $\mathcal{Y}_{2,s_2}$, and changing the global convex problem to a new $\mathcal{P}_s$ (purple region). The ADMM iterations continue, now with subsystem 2 having changed its local minimization problem (20a) by changing $\mathcal{Y}_2^{(s_2)}$. At the fourth iteration (purple dot),

Algorithm 2: D-rout (Dynamics Rollout).

Inputs: States $\{x_i\}_{i\in\mathcal{M}}$ and control sequences $\{\hat{\mathbf{u}}_i\}_{i\in\mathcal{M}}$
Initialize: $x_i(k) \leftarrow 0, x_j^{(i)}(k) \leftarrow 0, F_{i,\text{dr}} \leftarrow 0\ \forall k \in \mathcal{K}'\ \forall j \in \mathcal{N}_i\ \forall i \in \mathcal{M}$
for $k = 0, \dots, N-1$ **do**
  $\forall i \in \mathcal{M}$:
  $x_i(k) \leftarrow x_i$
  Transmit x_i to each neighbor $j \in \mathcal{N}_i$
  For each $j \in \mathcal{N}_i$: receive x_j and set $x_j^{(i)}(k) \leftarrow x_j$
  if $k < N$ **then**
    $F_{i,\text{dr}} \leftarrow F_{i,\text{dr}} + \ell_i(x_i, \hat{u}_i(k), \{x_j^{(i)}(k)\}_{j\in\mathcal{N}_i})$
    $x_i \leftarrow f_i(x_i, \hat{u}_i(k), \{x_j\}_{j\in\mathcal{N}_i})$
  else
    $F_{i,\text{dr}} \leftarrow F_{i,\text{dr}} + V_{\text{f},i}(x_i)$
  end if
end for
Outputs: $\{\mathbf{y}_{i,\text{dr}} = (\mathbf{x}_i^\top, \text{col}_{j\in\mathcal{N}_i}^\top(\mathbf{x}_j^{(i)}), \hat{\mathbf{u}}_i^\top)^\top, F_{i,\text{dr}}\}_{i\in\mathcal{M}}$

Algorithm 3: Sw-ADMM: Procedure for Solving $\mathcal{P}_\text{d}$.

Inputs: States $\{x_i\}_{i\in\mathcal{M}}$, initial guesses $\{\hat{\mathbf{u}}_i\}_{i\in\mathcal{M}}$, and initial local copies $\{\{\mathbf{x}_j^{(i)}\}_{j\in\mathcal{N}_i}\}_{i\in\mathcal{M}}$
Initialize: $\mathbf{z}_i \leftarrow 0$ and $\boldsymbol{\lambda}_i \leftarrow 0\ \forall i \in \mathcal{M}$
$S_i \leftarrow$ Eval-switching$(x_i, \hat{\mathbf{u}}_i, \{\mathbf{x}_j^{(i)}\}_{j\in\mathcal{N}_i})\ \forall i \in \mathcal{M}$
$s_i \leftarrow s_i' \in S_i$ and set local $\mathcal{Y}_i^{(s_i)}\ \forall i \in \mathcal{M}$
for $\tau = 0, \dots, T_\text{ADMM} - 1$ **do**
  $\forall i \in \mathcal{M}$:
  Get $\mathbf{y}_i^{\tau+1} = (\tilde{\mathbf{x}}_i^{\tau+1}, \mathbf{u}_i^{\tau+1})^\top$ from local minimization (20a), where $\tilde{\mathbf{x}}_i^{\tau+1} = ((\mathbf{x}_i^{\tau+1})^\top, \text{col}_{j\in\mathcal{N}_i}^\top(\mathbf{x}_j^{(i),\tau+1}))^\top$
  For each $j \in \mathcal{N}_i$: transmit $\mathbf{x}_j^{(i),\tau+1}$ to neighbor j
  For each $j \in \mathcal{N}_i$: receive $\mathbf{x}_i^{(j),\tau+1}$ from neighbor j
  Get $\mathbf{z}_i^{\tau+1}$ as (20b)
  Get $\boldsymbol{\lambda}_i^{\tau+1}$ as (20c)
  if $\tau < T_\text{cut}$ **then**
    $S_i \leftarrow$ Eval-switching$(x_i, \mathbf{u}_i^{\tau+1}, \{\mathbf{x}_j^{(i),\tau+1}\}_{j\in\mathcal{N}_i})$
    if $S_i \setminus \{s_i\} \neq \emptyset$ **then**
      $s_i \leftarrow s_i' \in S_i \setminus \{s_i\}$ and set local $\mathcal{Y}_i^{(s_i)}$
    end if
  end if
end for
Outputs: $\{\mathbf{y}_i^{T_\text{ADMM}}\}_{i\in\mathcal{M}}$

subsystem 1 identifies a new switching sequence $s_1 = (1, 2)$, and changes $\mathcal{Y}_1^{(s_1)}$, changing the global problem to the red region. Finally, as the local solutions are no longer pushed to the boundaries of the local constraint sets $\mathcal{Y}_i^{(s_i)}$, the iterations converge to the minimum of the convex piece (red region), a local minimum for $\mathcal{P}_\text{d}$.

Generating local switching sequences with Algorithm 1 requires a set of local copies $\{\mathbf{x}_j^{(i)}\}_{j\in\mathcal{N}_i}$. While these are an output of the local minimization step (20a), for the initial switching sequence, i.e., determining the yellow region from the blue dot in Fig. 1, these values can be obtained with the distributed procedure outlined in Algorithm 2. This procedure involves N steps of rolling out local dynamics under an initial guess for the control sequence, and neighbor-to-neighbor communications. Additional outputs of Algorithm 2 are the predicted local state trajectory $\mathbf{x}_i$ and the local cost F_i, under the initial control guess. These become useful in Section IV.

Algorithm 3 formalizes the switching ADMM procedure. As an input to Algorithm 3 a set of initial guesses $\{\hat{\mathbf{u}}_i\}_{i\in\mathcal{M}}$ is required that generates valid switching sequences in steps 3 and 4. Such control sequences are defined as follows.

Definition 2 (Weakly feasible local control sequences): The set of local control sequences $\{\mathbf{u}_i\}_{i\in\mathcal{M}}$ is *weakly feasible* for x if the corresponding global switching sequence $s = (s_1, \dots, s_M)$, generated in steps 3–4 of Algorithm 3, defines a feasible optimal control problem $\mathcal{P}_s(x)$, i.e., $\mathcal{Y}_s(x) \neq \emptyset$. Trivially, any globally feasible set of control sequences is also weakly feasible.

Note that generating weakly feasible control sequences is significantly easier than generating globally feasible control sequences, as weakly feasible control sequences do not need to satisfy the constraints in $\mathcal{P}_\text{d}(x)$, in particular the terminal constraints; rather, the only requirement is that they generate a global switching sequence s for which *there exists* a solution to the corresponding problem $\mathcal{P}_s(x)$. Construction of these guesses is discussed in Sections III-C and IV. Further, we highlight that in Algorithm 3 communication between subsystems occurs only in steps 7 and 8, where the communicated quantities are the predicted state trajectories $\mathbf{x}_i^{(j),\tau+1}$ for each $j \in \mathcal{N}_i$.

We introduce a cutoff number of iterations T_cut, after which the switching sequences s_i no longer change. This protects against the edge case where local solutions pull each other into new switching sequences with equal force, resulting in continuous oscillation between two convex regions of $\mathcal{P}_\text{d}$. We note that, in our experiments, the continuous switching case is observed only rarely, with the iterations otherwise converging to a problem $\mathcal{P}_s$ that contains a local minimum of $\mathcal{P}_\text{d}$.

We now give the convergence result for Algorithm 3. Following the introduction of T_cut an additional assumption is required to prove convergence of the algorithm.

Assumption 1: For all $x \in \mathcal{X}_0$ there does not exist a global switching sequence $s = (s_1, \dots, s_M)$ such that

$$(\mathcal{Y}_s(x) = \emptyset) \wedge \left(\mathcal{Y}_i(x_i) \cup \mathcal{Y}_i^{(s_i)} \neq \emptyset \quad \forall i \in \mathcal{M}\right). \tag{25}$$

Assumption 1 protects against the edge case where, when $\tau = T_\text{cut}$, the local systems are in a configuration of local switching sequences that represent a $\mathcal{P}_s$ with an empty feasible set. In addition, as the convergence of ADMM is an asymptotic result, we make the following assumption.

Assumption 2: $T_\text{ADMM} \to \infty$.

Proposition 2: For all $x \in \mathcal{X}_0$ and weakly feasible $\{\hat{\mathbf{u}}_i\}_{i\in\mathcal{M}}$, under Assumptions 1 and 2, the outputs of Algorithm 3, $\{\mathbf{y}_i^{T_\text{ADMM}}\}_{i\in\mathcal{M}}$, converge to a consensus point, i.e., $r(\mathbf{y}^{T_\text{ADMM}}) \to 0$ and $\mathbf{y}^{T_\text{ADMM}} \in \mathcal{Y}$.

Proof: For iterates $\tau \geq T_\text{cut}$ the local sets $\mathcal{Y}_i^{(s_i)}$ are fixed and, by Assumption 1, the corresponding $\mathcal{P}_s$ has a nonempty feasible

Algorithm 4: Distributed MPC, Executed Each Time Step.

1: Measure local states $x_i \; \forall i \in \mathcal{M}$
2: Generate initial weakly feasible guesses $\hat{\mathbf{u}}_i \; \forall i \in \mathcal{M}$
3: $\{\mathbf{y}_{i,\mathrm{dr}}, F_{i,\mathrm{dr}}\}_{i\in\mathcal{M}} \leftarrow$ D-rout$(\{x_i\}_{i\in\mathcal{M}}, \{\hat{\mathbf{u}}_i\}_{i\in\mathcal{M}})$
4: Solve $\mathcal{P}_\mathrm{d}(x)$ with Algorithm 3:
$\{\mathbf{y}_i = (\tilde{\mathbf{x}}_i^\top, \mathbf{u}_i^\top)^\top\}_{i\in\mathcal{M}} \leftarrow$
Sw-ADMM$(\{x_i\}_{i\in\mathcal{M}}, \{\hat{\mathbf{u}}_i\}_{i\in\mathcal{M}}, \{\{\mathbf{x}_j^{(i)}\}_{j\in\mathcal{N}_i}\}_{i\in\mathcal{M}})$
5: Apply local inputs $u_i(0) \; \forall i \in \mathcal{M}$

set. Therefore, as $\mathcal{P}_s$ is convex, the standard ADMM iterations (20) converge to a consensus point [9]. □

Remark 2: Assumption 1 is an assumption on the strength of the coupling between subsystems. Providing conditions to easily verify Assumption 1 is beyond the scope of this article. An alternative approach, avoiding Assumption 1, would be to permit subsystems to switch their local switching sequence only after the new global convex region has been checked for feasibility. This is left to future work.

Remark 3: While the convergence of ADMM, and hence Algorithm 3, is asymptotic, in practice ADMM often converges to an acceptable accuracy within a few iterations [9], motivating the use of a finite T_ADMM. Furthermore, if errors induced by finite termination were to become an issue, existing work that ensures feasibility and stability under finitely terminated distributed optimization [17], [19] can be added to our approach.

C. Distributed MPC Algorithm

Leveraging the switching ADMM procedure we now present the distributed MPC algorithm in Algorithm 4. In step 2, each subsystem generates a weakly feasible guess $\hat{\mathbf{u}}_i$. In Section IV, it is shown how subsystems can locally generate these guesses using a terminal control law. Here, however, we highlight that these initial guesses can often be generated locally using heuristics specific to the application. In addition, a common heuristic is to use the shifted solution from the previous time step, i.e., $\hat{\mathbf{u}}_i = (\bar{u}_i^\top(1), \ldots, \bar{u}_i^\top(N-2), \bar{u}_i^\top(N-2))^\top$, where $\bar{\mathbf{u}}_i = (\bar{u}_i^\top(0), \ldots, \bar{u}_i^\top(N-1))^\top$ is the solution of the previous time step. Finally, state constraints can be made soft constraints through the introduction of slack variables, penalized in the objective function, to ensure any initial feasible guess is globally, and hence also weakly, feasible.

As Algorithm 3 generates, in general, a local minimum of $\mathcal{P}_\mathrm{d}$, the quality of the solution could be improved by running it in parallel with different initial guesses, introducing an effective multistart approach. This is of course limited by the communication and computation limits of the subsystems.

IV. Stability and Recursive Feasibility

In this section, we give a stable and recursively feasible extension to Algorithm 4 under additional assumptions on the system. Given the local properties of Algorithm 3, we leverage a dual-mode approach to stability [13], [20], where terminal constraints are added to the MPC scheme and subsystems use stable switching linear controllers when they enter a terminal set. This introduces a complex tradeoff between the size of the terminal set, the length of the prediction horizon, and closed-loop performance. For a larger terminal set the systems use linear control laws for a larger portion of the state-space, using the optimization-based MPC control law less and foreseeably sacrificing performance. With a smaller terminal set, on the other hand, the MPC controller is used more, but requires a longer horizon to satisfy the terminal constraint. We note that, as Algorithm 3 involves only convex programs, long horizons present less computational demand than for approaches that rely on mixed-integer programming.

A. Assumptions

Define $\hat{\mathcal{L}}_i = \{l \in \mathcal{L}_i | 0 \in \overline{P}_i^{(l)}\}$ the set of subsystems i's PWA regions containing or touching the origin. We introduce the following extra assumptions on the systems considered.

Assumption 3 (See [13]): For all $i \in \mathcal{M}$ the matrix pairs $(A_i^{(l)}, B_i^{(l)}), \; \forall l \in \mathcal{L}_i$, are controllable; so for all $l \in \mathcal{L}_i$ there exists a gain $K_i^{(l)}$ such that $A_i^{(l)} + B_i^{(l)} K_i^{(l)}$ is a Schur matrix. The origin is an equilibrium point with $u_i = 0$ and $x_j = 0, \; \forall j \in \mathcal{N}_i$, i.e., $c_i^{(l)} = 0, \; \forall l \in \hat{\mathcal{L}}_i$.

Assumption 3 says that, for each subsystem, the origin is an equilibrium of the dynamics in all PWA regions that touch it, and that there exists a stabilizing linear feedback law within each of these regions.

Assumption 4: For all $i \in \mathcal{M}$ there exists a nonempty terminal set $\mathcal{X}_{\mathrm{T},i} \subseteq \bigcup_{l\in\hat{\mathcal{L}}_i} P_i^{(l)}$, containing the origin, such that $\forall x_i \in \mathcal{X}_{\mathrm{T},i}$ we have $f_i(x_i, u_i, \{x_j\}_{j\in\mathcal{N}_i}) \in \mathcal{X}_{\mathrm{T},i}, \; \forall x_j \in \mathcal{X}_{\mathrm{T},j}, \; \forall j \in \mathcal{N}_i$, with $u_i = K_i^{(l)} x_i$ and $K_i^{(l)} x_i \in \mathcal{U}_i$.

Define the global terminal set to be $\mathcal{X}_\mathrm{T} = \{x | x_i \in \mathcal{X}_{\mathrm{T},i}, \forall i \in \mathcal{M}\}$. Assumption 4 guarantees that there exist local terminal sets that are positively invariant under the switching terminal controllers and that are robust to the effects of coupling when neighboring subsystems are also within their terminal sets. This is a less restrictive assumption than that in [13], where the terminal sets are assumed to be robust to the effects of coupling for the entire state space of neighboring systems $\mathcal{X}_j$. In contrast to [13], this less restrictive assumption is possible due to the proposed approach explicitly giving agreement on the values of shared states over the prediction horizon, rendering the approach applicable to systems with stronger coupling. Note that, Ma et al. [13] proposed a method to compute the terminal sets required for the controller. These sets trivially satisfy Assumption 4 as $\mathcal{X}_{\mathrm{T},j} \subset \mathcal{X}_j$. In Appendix A, we present a modification to the procedure given in [13] to calculate less restrictive sets satisfying Assumption 4. In addition, we provide a formal proof, missing in [13], for the forward invariance property of the resulting sets. Clearly, the sets required by Ma et al. [13] do not exist for systems with unbounded state spaces, while the sets in Assumption 4 can still exist.

Assumption 5: The stage and terminal costs are of the form $\ell_i(x_i, u_i, \{x_j\}_{j\in\mathcal{N}_i}) = x_i^\top Q_i x_i + u_i^\top R u_i + \sum_{j\in\mathcal{N}_i} x_j^\top Q_{ij} x_j$ and $V_{\mathrm{f},i}(x) = x_i^\top \Phi_i x_i$ for positive definite Q_i, R_i, and Φ_i. The terminal costs $V_{\mathrm{f},i}$ and controllers $K_i^{(l)}$ are such that for all

$x \in \mathcal{X}_{\mathrm{T}}$

$$\sum_{i\in\mathcal{M}} \left(V_{\mathrm{f},i}\left(f_i(x_i,u_i,\{x_j\}_{j\in\mathcal{N}_i})\right) - V_{\mathrm{f},i}(x_i) + \ell_i(x_i,u_i,\{x_j\}_{j\in\mathcal{N}_i})\right) \leq 0 \tag{26}$$

where $u_i = K_i^{(l)}x_i, x_i \in P_i^{(l)}$.

Assumption 5 says that the terminal cost matrices Φ_i result in $\sum_{i\in\mathcal{M}} V_{\mathrm{f},i}$ being a global common quadratic Lyapunov function for all possible combinations of local PWA regions within the terminal set. The computation of such terminal costs has been covered for single PWA systems in [21] and for distributed linear systems in [22]. In Appendix B, we outline an LMI approach to constructing terminal costs $V_{\mathrm{f},i}$ that satisfy Assumption 5. While Assumption 5 may be restrictive if the origin is on the boundary of many PWA regions, note that we could restrict the local terminal sets to a single PWA region, $\mathcal{X}_{\mathrm{T},i} \subseteq P_i^{(l_i)}, l_i \in \hat{\mathcal{L}}_i$, such that finding a terminal cost satisfying Assumption 5 reduces to finding a Lyapunov function for the linear system defined by the PWA regions $\{P_i^{(l_i)}\}_{i\in\mathcal{M}}$, rather than a common Lyapnuov function over arbitrary switching. Again we note that, as Algorithm 3 involves only convex programs, long horizons are computationally tractable and smaller terminal sets are therefore viable.

Assumption 6: A weakly feasible initial set of control sequences is assumed to be known for the first time step.

We highlight that in the distributed MPC literature it is common to assume access to a *globally feasible* set of initial control sequences [23], [24], which is a much stronger assumption than Assumption 6.

B. Stabilizing Distributed MPC Algorithm

With a slight abuse of notation, we redefine the optimal control problems $\mathcal{P}_{\mathrm{d}}(x)$ and $\mathcal{P}_s(x)$ by modifying the feasible sets $\mathcal{Y}$ in (8) and $\mathcal{Y}_i$ in (17) to include terminal constraints

$$\begin{aligned}\mathcal{Y}(x) &= \{\mathbf{y} \,|\, (7b)-(7e), x_i(N) \in \mathcal{X}_{\mathrm{T},i}, \forall i \in \mathcal{M}\}\\ \mathcal{Y}_i(x_i) &= \{\mathbf{y}_i \,|\, (7c),(7d), x_i(N) \in \mathcal{X}_{\mathrm{T},i}\}\end{aligned} \tag{27}$$

naturally redefining $\mathcal{X}_0 = \{x \mid \mathcal{Y}(x) \neq \emptyset\}$. We then introduce the stabilizing distributed MPC algorithm in Algorithm 5.

When all subsystems are in the terminal regions the switching linear feedback control laws are used. Otherwise, for Algorithm 3, subsystems have local access to a feasible initial guess through shifting the previous control solution and adding the terminal controller as the final element of the initial feasible guess. Steps 13−15 provide a verification that the solution generated by the switching ADMM procedure has improved the value with respect to the initial guesses. The verification is local, as both $F_i(\mathbf{x}_i,\mathbf{u}_i,\{\mathbf{x}_j^{(i)}\}_{j\in\mathcal{N}_i})$ and $F_{i,\mathrm{dr}}$ are known by subsystem i. This protects against the edge case where the local minimum found by the procedure is worse than the initialization.

We now prove the recursive feasibility and the closed-loop stability of Algorithm 5.

Theorem 1: Assume Assumptions 1−6 hold. For an initial state $x \in \mathcal{X}_0 \setminus \mathcal{X}_{\mathrm{T}}$, at time step $t = 0$, a weakly feasible set of initial control guesses $\{\hat{\mathbf{u}}_i\}_{i\in\mathcal{M}}$, and generating control inputs via Algorithm 5, $\mathcal{P}_{\mathrm{d}}(x)$ is feasible for all time steps $t \geq 0$ in which $x \notin \mathcal{X}_{\mathrm{T}}$.

Algorithm 5: Stable Distributed MPC, Executed Each Time Step.

1: Measure local states $x_i\ \forall i \in \mathcal{M}$
2: **if** $x_i \in \mathcal{X}_{\mathrm{T},i}\ \forall i \in \mathcal{M}$
3: Apply local inputs $u_i = K_i^{(l)}x_i, x_i \in P_i^{(l)}\ \forall i \in \mathcal{M}$
4: **else**
5: **if** first time step **then**
6: $\hat{\mathbf{u}}_i \leftarrow$ known feasible guess $\forall i \in \mathcal{M}$
7: **else**
8: Construct initial guess from terminal controller and previous solution $\hat{\mathbf{u}}_i \leftarrow (\bar{u}_i^\top(1),\ldots,\bar{u}_i^\top(N-1),(K_i^{(l)}\bar{x}_i(N))^\top)^\top, \bar{x}_i(N) \in P_i^{(l)}\ \forall i \in \mathcal{M}$
9: **end if**
11: $\{\mathbf{y}_{i,\mathrm{dr}}, F_{i,\mathrm{dr}}\}_{i\in\mathcal{M}} \leftarrow$ D-rout$(\{x_i\}_{i\in\mathcal{M}}, \{\hat{\mathbf{u}}_i\}_{i\in\mathcal{M}})$
12: Solve $\mathcal{P}_{\mathrm{d}}(x)$ with Algorithm 3: $\{\mathbf{y}_i\}_{i\in\mathcal{M}} \leftarrow$ Sw-ADMM$(\{x_i\}_{i\in\mathcal{M}}, \{\hat{\mathbf{u}}_i\}_{i\in\mathcal{M}}, \{\{\mathbf{x}_j^{(i)}\}_{j\in\mathcal{N}_i}\}_{i\in\mathcal{M}})$
13: **if** not $(F_i(\mathbf{x}_i,\mathbf{u}_i,\{\mathbf{x}_j^{(i)}\}_{j\in\mathcal{N}_i}) \leq F_{i,\mathrm{dr}}\ \forall i \in \mathcal{M})$ **then**
14: $\mathbf{y}_i \leftarrow \mathbf{y}_{i,\mathrm{dr}}\ \forall i \in \mathcal{M}$
15: **end if**
16: $\bar{\mathbf{x}}_i \leftarrow \mathbf{x}_i$ and $\bar{\mathbf{u}}_i \leftarrow \mathbf{u}_i\ \forall i \in \mathcal{M}$
17: Apply local inputs $u_i(0)\ \forall i \in \mathcal{M}$
18: **end if**

Proof: The proof works by induction. Assume that, for state $x \notin \mathcal{X}_{\mathrm{T}}$, at time step t, $\mathcal{P}_{\mathrm{d}}(x)$ is feasible, and that subsystems have weakly feasible initial control guesses $\hat{\mathbf{u}}_i, \forall i \in \mathcal{M}$. By Proposition 2, with weakly feasible initial guesses, the outputs $\mathbf{y}_i = (\tilde{\mathbf{x}}_i^\top, \mathbf{u}_i^\top)^\top$ of Algorithm 3 are feasible for $\mathcal{P}_{\mathrm{d}}(x)$, and therefore $x_i(N) \in \mathcal{X}_{\mathrm{T},i}, \forall i \in \mathcal{M}$. The subsystems apply local inputs $u_i(0)$, propagating the state dynamics as $x_i^+ = f_i(x_i,u_i(0),\{x_j\}_{j\in\mathcal{N}_i})$ and, as by Proposition 2 $\mathbf{x}_j^{(i)} = \mathbf{x}_j, \forall j \in \mathcal{N}_i$, the new states coincide with the first predicted state, i.e., $x_i^+ = x_i(1)$. At the next time step $t+1$, in state x^+, the initial guesses are $\hat{\mathbf{u}}_i = (u_i^\top(1),\ldots,u_i^\top(N-1),(K_i^{(l)}x_i(N))^\top)^\top, x_i(N) \in P_i^{(l)} \cap \mathcal{X}_{\mathrm{T},i},\ i \in \mathcal{M}$, and are globally feasible, and therefore also weakly feasible, as $\mathcal{X}_{\mathrm{T},i}$ is forward invariant and robust to coupling under the terminal controllers when $x_j \in \mathcal{X}_{\mathrm{T},j}, \forall j \in \mathcal{N}_i$. Therefore $\mathcal{P}_{\mathrm{d}}(x^+)$ is feasible at time step $t+1$.

Now, for time step $t = 0$ and initial state $x \in \mathcal{X}_0 \setminus \mathcal{X}_{\mathrm{T}}$, $\mathcal{P}_{\mathrm{d}}(x)$ is feasible as $x \in \mathcal{X}_0$, and the initial guesses $\{\hat{\mathbf{u}}_i\}_{i\in\mathcal{M}}$ for Algorithm 3 are weakly feasible. It follows then by induction that $\mathcal{P}_{\mathrm{d}}(x)$ is feasible for all time steps $t > 0$ in which $x \notin \mathcal{X}_{\mathrm{T}}$. □

Remark 4: Theorem 1 considers only states $x \notin \mathcal{X}_{\mathrm{T}}$ as $\mathcal{P}_{\mathrm{d}}$ is not relevant in Algorithm 5 when $x \in \mathcal{X}_{\mathrm{T}}$.

Lemma 2: Assume Assumptions 1−6 hold. For all $x \in \mathcal{X}_0 \setminus \mathcal{X}_{\mathrm{T}}$ and time steps $t > 0$, generating control inputs via Algorithm 5 we have

$$V(x,\mathbf{u}) \leq V(x,\hat{\mathbf{u}}) \tag{28}$$

where $\hat{\mathbf{u}} = (\hat{\mathbf{u}}_1^\top, \ldots, \hat{\mathbf{u}}_M^\top)^\top$ is the initial guess for Algorithm 3 (step 12 of Algorithm 5), and $\mathbf{u}$ is the resulting global control input sequence (step 15 of Algorithm 5), with V defined in (5).

Proof: For time steps $t > 0$, $\hat{\mathbf{u}}$ is globally feasible, i.e., $V(x, \hat{\mathbf{u}}) < \infty$. Observe that $\sum_{i\in\mathcal{M}} F_{i,\text{dr}} = V(x, \hat{\mathbf{u}})$. If there exists $i \in \mathcal{M}$ such that $F_i(\mathbf{x}_i, \mathbf{u}_i, \{\mathbf{x}_j^{(i)}\}_{j\in\mathcal{N}_i}) > F_{i,\text{dr}}$ in step 13, then the input $\mathbf{u}$ is set to $\hat{\mathbf{u}}$, and $V(x, \mathbf{u}) = V(x, \hat{\mathbf{u}})$. If $F_i(\mathbf{x}_i, \mathbf{u}_i, \{\mathbf{x}_j^{(i)}\}_{j\in\mathcal{N}_i}) \leq F_{i,\text{dr}}$, $\forall i \in \mathcal{M}$ in step 13, then $\sum_{i\in\mathcal{M}} F_i(\mathbf{x}_i, \mathbf{u}_i, \{\mathbf{x}_j^{(i)}\}_{j\in\mathcal{N}_i}) \leq V(x, \hat{\mathbf{u}})$. By Proposition 2 $\mathbf{x}_j^{(i)} = \mathbf{x}_j, \forall j \in \mathcal{N}_i$, and therefore the summation is equal to $V(x, \mathbf{u})$, i.e., $V(x, \mathbf{u}) \leq V(x, \hat{\mathbf{u}})$. □

Theorem 2: Assume Assumptions 1–6 hold. The control law generated by Algorithm 5 is exponentially stabilizing with a region of attraction $\mathcal{X}_0$.

Proof: Take $\mathbf{u} = \{\mathbf{u}_i\}_{i\in\mathcal{M}}$ as the global input generated in Algorithm 5 at time step t with global state $x \in \mathcal{X}_0 \setminus \mathcal{X}_\text{T}$. Furthermore, take $\{x_i(N)\}_{i\in\mathcal{M}}$ as the set of predicted states at time step $t + N$ after applying the control inputs $\mathbf{u}$. By Proposition 2 we have $x_i(N) \in \mathcal{X}_{\text{T},i}, \forall i \in \mathcal{M}$. Take $\hat{\mathbf{u}}_i = (u_i^\top(1), \ldots, u_i^\top(N-1), (K_i^{(l)} x_i(N))^\top)^\top, x_i(N) \in P_i^{(l)}$ as the globally feasible initial guess at the next time step $t + 1$, constructed from $\mathbf{u}_i$, $x^+ \in \mathcal{X}_0 \setminus \mathcal{X}_\text{T}$ the global state at time step $t + 1$ after applying local inputs $u_i(0)$, and $\mathbf{u}^+$ as the global input generated in Algorithm 5 at time step $t + 1$. Consider the values $V(x^+, \hat{\mathbf{u}}) < \infty$ and $V(x, \mathbf{u}) < \infty$. As the control sequence $\hat{\mathbf{u}}$ is a shifted version of $\mathbf{u}$, the resulting state trajectories will overlap except for the first and last time steps. We thus have the relation [8]

$$\begin{aligned} V(x^+, \hat{\mathbf{u}}) - V(x, \mathbf{u}) = &- \sum_{i\in\mathcal{M}} \ell_i\left(x_i, u_i(0), \{x_j\}_{j\in\mathcal{N}_i}\right) \\ &+ \sum_{i\in\mathcal{M}} \Bigg(V_{\text{f},i}\left(f_i\left(x_i(N), K_i^{(l)} x_i(N), \{x_j(N)\}_{j\in\mathcal{N}_i}\right)\right) \\ &- V_{\text{f},i}\left(x_i(N)\right) + \ell_i\left(x_i(N), K_i^{(l)} x_i(N), \{x_j(N)\}_{j\in\mathcal{N}_i}\right)\Bigg). \end{aligned} \tag{29}$$

By Assumption 5 the second summation term is nonpositive, thus $V(x^+, \hat{\mathbf{u}}) - V(x, \mathbf{u}) \leq -\sum_{i\in\mathcal{M}} \ell_i(x_i, u_i(0), \{x_j\}_{j\in\mathcal{N}_i})$. Further, the conditions of Lemma 2 hold at time step $t + 1$, such that $V(x^+, \mathbf{u}^+) \leq V(x^+, \hat{\mathbf{u}})$, giving the expression

$$\begin{aligned} V(x^+, \mathbf{u}^+) - V(x, \mathbf{u}) &\leq V(x^+, \hat{\mathbf{u}}) - V(x, \mathbf{u}) \\ &\leq -\sum_{i\in\mathcal{M}} \ell_i\left(x_i, u_i(0), \{x_j\}_{j\in\mathcal{N}_i}\right) \end{aligned} \tag{30}$$

for all $x \in \mathcal{X}_0 \setminus \mathcal{X}_\text{T}$. Assumption 5 on ℓ_i guarantees the existence of a class-$\mathcal{K}$ function $\beta(\cdot)$ such that

$$\sum_{i\in\mathcal{M}} \ell_i(x_i, u_i, \{x_j\}_{j\in\mathcal{N}_i}) \geq \beta(\|(x, u)\|) \tag{31}$$

for all $x \notin \mathcal{X}_T$, and for all $u \in \mathcal{U}_1 \times \cdots \times \mathcal{U}_M$. As $\mathcal{X}_\text{T}$ contains the origin there exists $b > 0$ such that $V(x^+, \mathbf{u}^+) - V(x, \mathbf{u}) \leq -b$ for all $x \notin \mathcal{X}_\text{T}$.

For initial state $x \in \mathcal{X}_0 \setminus \mathcal{X}_\text{T}$, at time step $t = 0$, we have $V(x, \mathbf{u}) < \infty$ by Assumption 6. Let $x^{+\bar{t}}$ and $\mathbf{u}^{+\bar{t}}$ denote the global state and global control input sequence at time step $\bar{t}$, with $\bar{t}$ a finite integer such that $(\bar{t} - 1)b > V(x, \mathbf{u})$. If $x^{+\bar{t}} \notin \mathcal{X}_\text{T}$, then $V(x^+, \mathbf{u}^+) - V(x, \mathbf{u}) \leq -b$ holds for all time steps $t = 0, \ldots, \bar{t} - 1$. It follows that $V(x^{+\bar{t}}, \mathbf{u}^{+\bar{t}}) \leq V(x, \mathbf{u}) - (\bar{t} - 1)b < 0$, which is a contradiction as the value V is nonnegative. There therefore exists, for any initial state $x \in \mathcal{X}_0 \setminus \mathcal{X}_\text{T}$, a finite time $\bar{t}$ at which the global state enters the global terminal set $x \in \mathcal{X}_\text{T}$ [20].

As all local systems use the terminal controllers $u_i = K_i^{(l)} x_i, x_i \in P_i^{(l)}$ for $x_i \in \mathcal{X}_{\text{T},i}$, and the local terminal sets are robustly positively invariant under this terminal control, we have $x \in \mathcal{X}_\text{T}$ for all $t \geq \bar{t}$. Finally, by Assumption 5, under the switching linear controllers there exists a global common quadratic Lyapunov function for all possible combinations of local PWA regions when the global state $x \in \mathcal{X}_\text{T}$. A common quadratic Lyapunov function implies exponential stability under arbitrary switching [25], hence the global system is exponentially stable under the terminal controllers. Therefore as, for any initial state $x \in \mathcal{X}_0$, the global state enters the terminal region $\mathcal{X}_\text{T}$ in finite time and is exponentially stable within $\mathcal{X}_\text{T}$, the origin is exponentially stable under Algorithm 5 with regions of attraction $\mathcal{X}_0$. □

V. Illustrative Examples

In this section, we provide two examples demonstrating our approach. The first considers the network from [13], satisfying Assumptions 3–6. This example demonstrates the stabilizing properties of Algorithm 5 and compares the approach to that of [13]. The second example considers the more realistic control challenge of a platoon of vehicles, proposed as a distributed hybrid MPC benchmark in [26], and demonstrates the efficacy of Algorithm 4 as a practical control scheme. All examples are simulated on an 11th Gen Intel laptop with four i7 cores, 3.00 GHz clock speed, and 16Gb of RAM. Source code for the simulations can be found online.[2]

A. Stabilizing Example

In the following, we refer to Algorithm 5 as SwA and the approach based on robust controllable sets presented in [13] as RCS. We consider also a centralized MPC controller that solves $\mathcal{P}$ directly as a MIQP, referred to as C. Consider the network of three PWA systems, from [13], with identical dynamics defined over $L = 4$ regions, as shown in Fig. 2. The systems' dynamics parameters, for $i = 1, 2, 3$, are

$$\begin{aligned} A_i^{(1)} = A_i^{(3)} &= \begin{bmatrix} 0.6324 & 0.2785 \\ 0.0975 & 0.5469 \end{bmatrix} \\ A_i^{(2)} = A_i^{(4)} &= \begin{bmatrix} 0.6555 & 0.7060 \\ 0.1712 & 0.0318 \end{bmatrix} \\ B_i^{(1)} = B_i^{(2)} = B_i^{(3)} = B_i^{(4)} &= \begin{bmatrix} 1 \\ 0 \end{bmatrix} \end{aligned} \tag{32}$$

[2]https://github.com/SamuelMallick/distributed-mpc-pwa,stable-dmpc-pwa/tree/paper_2024,hybrid-vehicle-platoon/tree/paper-2024

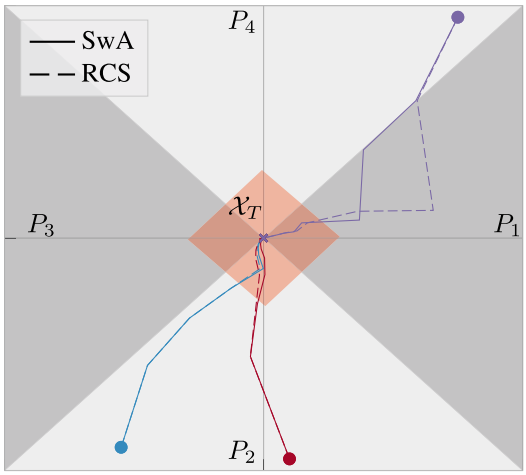

Fig. 2. State trajectories for SwA and RCS with weak coupling.

with state and action constraints $\mathcal{X}_i = \{x_i \mid |x_i|_\infty \leq 20\}$ and $\mathcal{U}_i = \{u_i \mid |u_i| \leq 3\}$, and stage costs $\ell_i = x_i^\top Q x_i + u_i^\top R u_i$ with

$$Q_i = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}, \quad R = 0.2. \tag{33}$$

The coupling in the network is

$$A_{12} = A_{21} = A_{23} = A_{31} = 2 \cdot 10^{-3} \cdot \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{34}$$

with all other coupling matrices zero. For a fair comparison[3] we use the same terminal controllers as [13], i.e., for $i = 1, 2, 3$

$$\begin{aligned} K_i^{(1)} = K_i^{(3)} &= \begin{bmatrix} -0.0544 & -0.1398 \end{bmatrix} \\ K_i^{(2)} = K_i^{(4)} &= \begin{bmatrix} -0.1544 & -0.0295 \end{bmatrix} \end{aligned} \tag{35}$$

and the same robustly positive invariant terminal sets, i.e., for $i = 1, 2, 3$

$$\mathcal{X}_{\mathrm{T},i} = \left\{ x \,\middle|\, \begin{bmatrix} P \\ -P \end{bmatrix} x \leq \gamma \mathbf{1} \right. \\ \left. P = \begin{bmatrix} 7.8514 & 8.1971 \\ 8.1957 & -7.8503 \end{bmatrix}, \gamma = 47 \right\}. \tag{36}$$

Finally, we compute local terminal costs $V_{\mathrm{f},i}(x_i) = x_i^\top \Phi_i x_i$, satisfying Assumption 5, using the procedure in Appendix B, which gives

$$\Phi_1 = \begin{bmatrix} 12.67 & 8.87 \\ 8.87 & 8.14 \end{bmatrix} \cdot 10^{-4} + \Phi$$

$$\Phi_2 = \begin{bmatrix} 10.58 & 7.90 \\ 7.90 & 8.26 \end{bmatrix} \cdot 10^{-4} + \Phi$$

$$\Phi_3 = \begin{bmatrix} 8.53 & 5.43 \\ 5.43 & 7.10 \end{bmatrix} \cdot 10^{-4} + \Phi$$

$$\Phi = \begin{bmatrix} 3.938 & 1.262 \\ 1.262 & 4.346 \end{bmatrix}. \tag{37}$$

For the SwA approach the horizon is $N = 5$. Algorithm 3 is run with $T_{\mathrm{ADMM}} = 50$ and $\rho = 0.5$. These values are hand chosen to ensure the total residual is less than 0.01. For this system the edge case of continuous switching is never observed; T_{cut} is consequently set to also be 50. For the weakly feasible guesses in the first time step, the trivial guesses $\hat{\mathbf{u}}_i = (0, \ldots, 0)^\top$ can be used. These trivial guesses are weakly feasible but not globally feasible, demonstrating the practical satisfaction of Assumption 6.

Fig. 2 shows the trajectories under each approach for the initial condition, $x_1(0) = \begin{bmatrix} -11 & -18 \end{bmatrix}^\top$, $x_2(0) = \begin{bmatrix} 2 & -19 \end{bmatrix}^\top$, $x_3(0) = \begin{bmatrix} 15 & 19 \end{bmatrix}^\top$. Both approaches drive the states to the origin. Fig. 3 shows, for 100 randomly sampled initial conditions, the relative performance drop with respect to the centralized controller $J_o / J_{\mathrm{C}}, o \in \{\mathrm{SwA}, \mathrm{RCS}\}$ with

$$J_o = \sum_{t=0}^{30} \sum_{i=1}^{3} \ell_i \left(x_i(t), u_i(t), \{x_j(t)\}_{j \in \mathcal{N}_i} \right). \tag{38}$$

The SwA approach generally performs slightly better, in terms of (38), than the RCS approach, with occasional outlier initial conditions for which the performance improvement is much more significant.[4] There is one outlier in which the RCS approach performs better, even outperforming the centralized controller, which is indeed possible with a finite prediction horizon.

Fig. 4 compares the online computation time required over all simulations for each approach using a range of solvers. The SwA approach allows the use of quadratic programming solvers, while the RCS approach requires MIQP solvers. The use of quadratic programming solvers allows the SwA approach to be much faster than the RCS approach, despite its iterative nature. The computation times for the centralized controller, solved using Gurobi [27], are in the order of tens of seconds, and are not included in Fig. 4 for clarity.

Now consider the same network, with the same terminal controllers, but with significantly higher coupling

$$A_{12} = A_{21} = A_{23} = A_{31} = \begin{bmatrix} 0.16 & 0 \\ 0 & 0.16 \end{bmatrix}. \tag{39}$$

The uncontrolled global system is now unstable. Under this coupling there do not exist local terminal sets that are robust to the coupling effects over the entire neighboring state spaces, and the RCS approach is hence not applicable. There do, however,

[3] In [13], a small disturbance is added to the dynamics and is accounted for in the computation of the robustly controllable sets. The approach in [13] requires no extra mechanism to account for the disturbance as the coupling between subsystems is also considered as a local disturbance. For a fair comparison we recompute all the sets required for the RCS approach with no additional disturbance in the dynamics.

[4] While unlikely in practical problems, in theory Algorithm 3 can converge to an arbitrarily bad local minimum. This is explored with randomized systems in Appendix C.

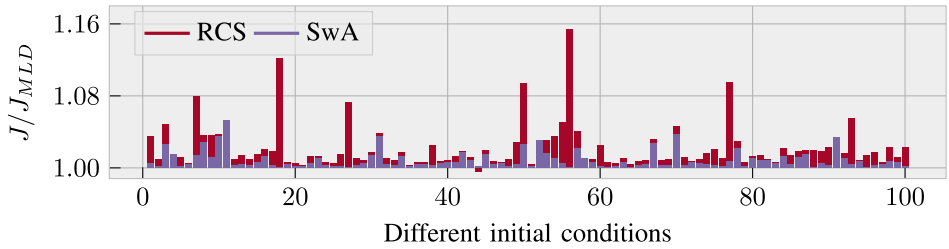

Fig. 3. Performance drop with respect to the centralized controller for 100 different initial conditions.

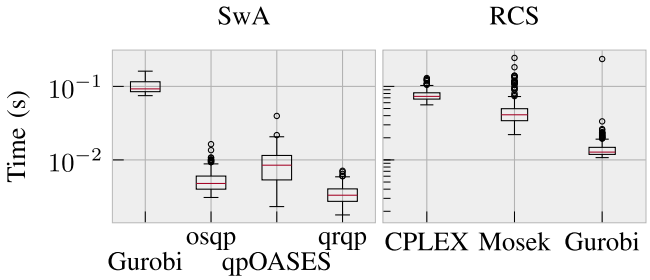

Fig. 4. Computation times with different solvers over 100 different initial conditions. The quadratic programming solvers are *osqp* [28], *qpOASES* [29], and *qrqp* [30]. The MIQP solvers are *CPLEX* [31], *Mosek* [32], and *Gurobi* [27].

exist terminal sets satisfying Assumption 4. Using the procedure in Appendix A, initialized with the sets in (36), we determine that the terminal sets in (36) satisfy Assumption 4 under the stronger coupling (39). Using the procedure in Appendix B the terminal costs $V_{\mathrm{f},i}(x_i) = x_i^\top \Phi_i x_i$ are computed as

$$\Phi_1 = \begin{bmatrix} 40.98 & 28.29 \\ 28.29 & 43.73 \end{bmatrix} \Phi_2 = \begin{bmatrix} 32.07 & 20.90 \\ 20.90 & 35.91 \end{bmatrix}$$
$$\Phi_3 = \begin{bmatrix} 31.97 & 20.83 \\ 20.83 & 35.07 \end{bmatrix}. \tag{40}$$

For the system with the increased coupling (39) the continuous switching case is observed to occur rarely. Algorithm 3 is hence run with $T_{\mathrm{ADMM}} = 75$, $T_{\mathrm{cut}} = 50$, and $\rho = 5$, with these values again hand chosen to ensure that the total residual is less than 0.01. Fig. 5 shows a closed-loop trajectory under this stronger coupling, demonstrating how our approach can provide stabilizing control for stronger coupling, due to its explicit handling of shared states and consequently less strict requirements on the terminal sets. Furthermore, Fig. 6 demonstrates the convergence of Algorithm 3 in the third time step when the continuous switching edge case occurs. Subsystem 1 continuously switches its local switching sequence, causing the local input and global residual to not converge. For iterations larger than $T_{\mathrm{cut}} = 50$ the local switching sequence does not change, and the local input and global residual converge.

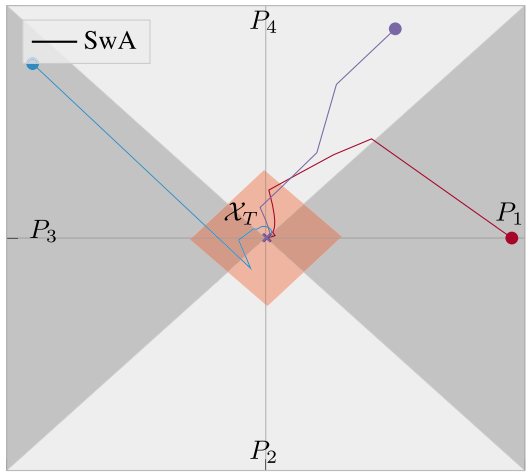

Fig. 5. State trajectories for SwA with strong coupling.

B. Hybrid Vehicle Platooning

We now consider the problem of hybrid vehicle platooning that was presented as a benchmark problem for hybrid distributed MPC methods in [26]. This example is a tracking problem and does not satisfy Assumptions 3–5; therefore, the approach in [13] cannot be used. Consider M vehicles that must form a platoon configuration while traveling in a single lane. Each vehicle i has state $x_i = \begin{bmatrix} p_i & v_i \end{bmatrix}^\top$, where $p_i \in \mathbb{R}$ and $v_i \in \mathbb{R}$ are the position and velocity of the vehicle, and control $u_i \in \mathbb{R}$, the normalized throttle position. The PWA vehicle dynamics $x_i^+ = f_i(x_i, u_i)$ have seven PWA regions, modeling discrete gear changes that switch dependent on velocity, changing the traction force generated by the vehicle's control input. For the sake of space the reader is referred to [26] for the full dynamics. The control goal is for each vehicle to maintain a desired separation from the vehicle in front of it while the front vehicle follows a reference trajectory. Without loss of generality it is assumed that the set of vehicles $\mathcal{M}$ is ordered by the positions of vehicles, such that vehicle 1 is the front vehicle, and vehicle i tracks the trajectory of vehicle $i-1$. The stage costs are then

$$\begin{aligned} \ell_1(x_1, u_1) &= \|x_i - r - \eta\|_Q + \|u_i\|_R \\ \ell_i(x_i, u_i, x_{i-1}) &= \|x_i - x_{i-1} - \eta\|_Q + \|u_i\|_R \end{aligned} \tag{41}$$

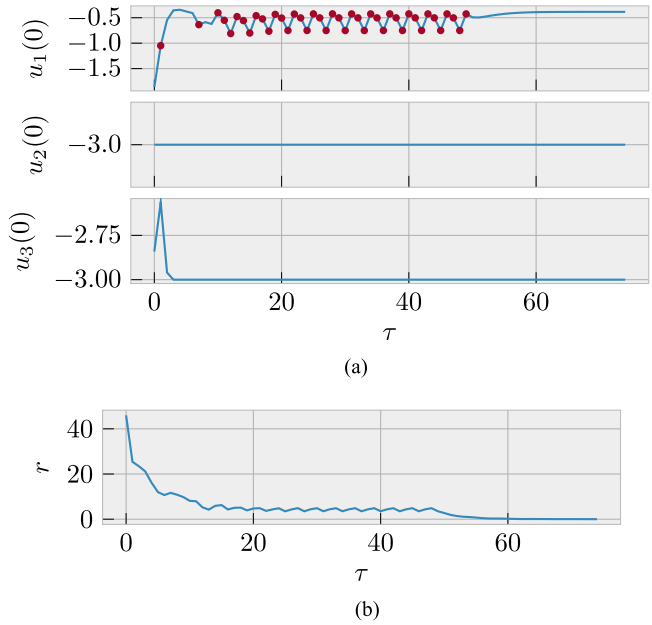

Fig. 6. Demonstration of the effect of T_{cut} in Algorithm 3. (a) Control inputs. Red dots indicate when a local switch of s_i occurred. (b) Residual.

for $i = 2, \dots, M$ with $Q = \begin{bmatrix} 1 & 0 \\ 0 & 0.1 \end{bmatrix}$, $R = 1$, $\eta = \begin{bmatrix} 50 & 0 \end{bmatrix}^\top$ and $r \in \mathbb{R}^2$ the reference state. Velocity and position bounds

$$3.94 \leq v_i \leq 45.84$$
$$0 \leq p_i \leq 10000 \tag{42}$$

define local convex constraints on states and inputs, and the coupling constraints

$$h_i(x_i, x_{i-1}) = x_i - x_{i-1} + d_{\text{safe}} \leq 0 \tag{43}$$

enforce a safe distance $d_{\text{safe}} = 25$ between vehicles. Vehicles are initialized with uniformly randomized velocities in the range $\begin{bmatrix} 5 & 30 \end{bmatrix}$ and with uniformly randomized positions in the range $\begin{bmatrix} 50 & 100 \end{bmatrix}$ behind the preceding vehicle.

We compare our approach against existing controllers outlined in [26]. For our approach, referred to as SwA, we again use a horizon of $N = 5, T_{\text{ADMM}} = 100$ and $\rho = 0.5$ for Algorithm 3, again ensuring the total residual is less than 0.01. Once again the edge case of continuous switching is never observed; T_{cut} is consequently set to also be 100. We implement a multistart approach with two initial guesses, choosing the solution that yields a lower cost. At time step t the shifted solution from time step $t-1$ is used, appending a constant control value

$$\mathbf{u}_i = \left(\bar{u}_i^\top(1), \dots, \bar{u}_i^\top(N-1), \bar{u}_i^\top(N-1)\right)^\top \tag{44}$$

as well as a constant control sequence, maintaining the current velocity v_i. The MPC controllers for comparison from [26] are based on converting the PWA dynamics to MLD form [10] and solving MIQPs. The centralized controller, referred to as C, solves $\mathcal{P}_{\text{d}}$ directly as one MIQP, considering all vehicles in one optimization problem, solving for all control inputs while considering all couplings. The sequential controller, referred to as S, is a distributed controller where each vehicle solves a local MIQP once, in a predefined order, communicating the solutions down the sequence of vehicles. Finally, the nonconvex ADMM controller, referred to as NCA, applies ADMM directly to $\mathcal{P}_{\text{d}}$, with vehicles iteratively solving local MIQPs. For NCA 100 iterations are used for a fair comparison between the two ADMM-based approaches. Optimizing the solver choice for computation time, for the SwA approach the quadratic programs are solved with the qpOASES solver [29], while all MIQPs are solved with Gurobi [27].

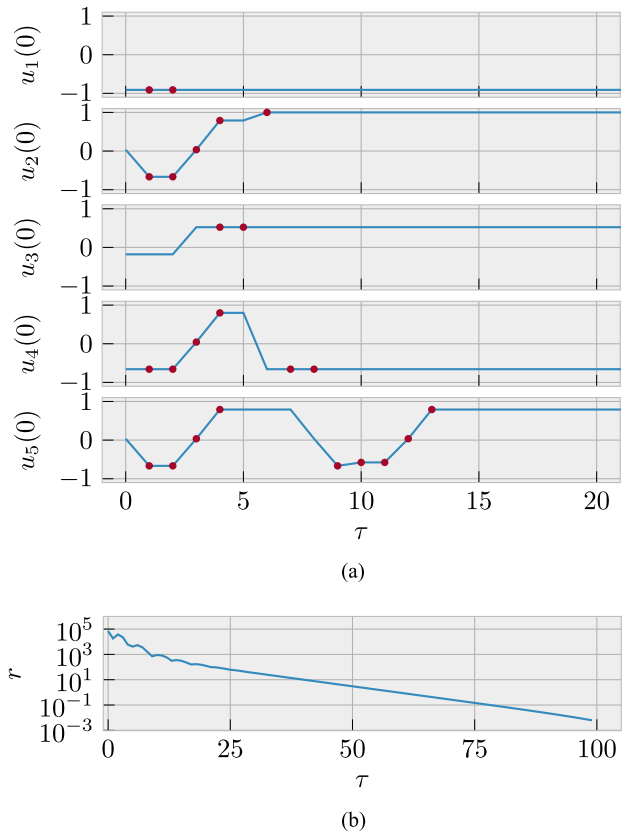

Fig. 7. Iterations of Algorithm 3. (a) Control inputs. Red dots indicate when a local switch of s_i occurred. (b) Residual (log scale).

To further illustrate the mechanism of Algorithm 3 Fig. 7(a) shows the local solutions $u_i(0)$ throughout the first 20 iterations[5] at the first time step $t = 0$ of a simulation with $M = 5$. Red dots indicate iterations, in which local systems changed to new switching sequences, changing the global approximation $\mathcal{P}_s$. It is seen that the inputs vary sharply as vehicles iteratively solve their local minimization problems, communicate solutions, and identify new local switching sequences. Eventually the changing of switching sequences stops, indicating that the local solutions are no longer pushed to the boundaries of the local constraint sets, and the solutions have settled in a local minimum of $\mathcal{P}_{\text{d}}$. Fig. 7(b) shows the total residual over the entire 100 iterations,

[5]For the remaining 80 iterations the values change negligibly.

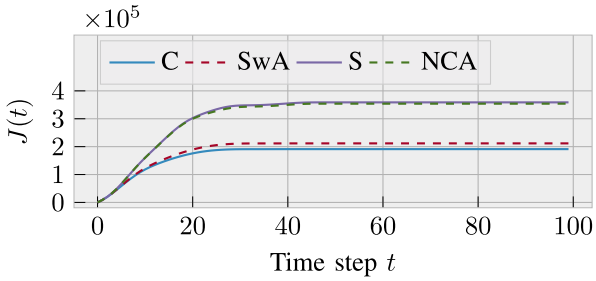

Fig. 8. Cumulative tracking cost for $M = 10$.

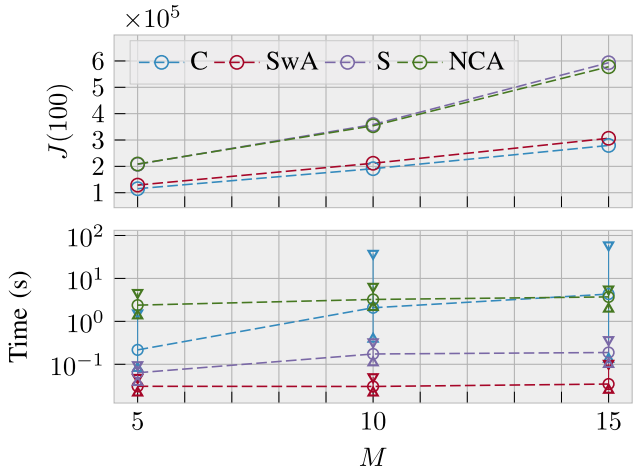

Fig. 9. Tracking cost (top) and max/average/min computation time (bottom) as M increases.

demonstrating that subsystems come to agree on the values of the coupled states, as in Theorem 1.

Fig. 8 shows, for $M = 10$ vehicles, the cumulative tracking cost

$$J(t) = \sum_{\tau=0}^{t} \ell_1\left(x_1(\tau), u_1(\tau)\right) + \sum_{i\in\mathcal{M}\setminus\{1\}} \ell_i\left(x_i(\tau), u_i(\tau), x_{i-1}(\tau)\right) \quad (45)$$

under each of the different controllers. All controllers manage to reach a stable platoon formation, where the tracking cost no longer grows. The centralized controller naturally performs the best as it finds the global optimum of $\mathcal{P}_\mathrm{d}$. The SwA controller outperforms both other distributed control solutions. Fig. 9 shows the total tracking cost and the computation times for each controller with $M = 5, 10$, and 15 vehicles. As the number of vehicles increases the performance of the sequential and nonconvex ADMM controllers decreases significantly with respect to the centralized controller, while the performance drop of the SwA controller scales much better. The computation times show that the SwA is the fastest controller as it involves the solutions to only QPs rather than MIQPs. Both the NCA and SwA approaches have computation times that scale elegantly with M, as the size of the optimization problems and the number of iterations is independent of M. However, as NCA solves MIQPs iteratively rather than QPs, it has a significantly higher computational burden.

VI. Conclusion

In this article, we presented a new approach for distributed MPC for PWA systems. The proposed controller is based on a novel switching ADMM procedure that solves a globally formulated nonconvex optimal control problem, without requiring mixed-integer programming, and provably giving agreement on the values of shared states. The recursive feasibility and stability of the resulting control scheme is proven under additional assumptions on the system. The theoretical properties of the approach have been demonstrated on a small numerical example, whilst the practical utility has been demonstrated on a larger, more realistic, hybrid control problem.

Future work will look at bounding the level of suboptimality between the closed loop performance of the proposed controller and the theoretically optimal performance of a centralized mixed-integer-based MPC controller.

Appendix A
Computation of Terminal Sets in Assumption 4

The procedure presented in [13] computes local positively invariant sets that are robust to the effects of coupling from neighboring subsystems. The procedure involves, for each subsystem i, computing a sequence of sets, beginning from a *reasonably large polytope* $\mathcal{X}_{0,i} \subseteq \bigcup_{l\in\hat{\mathcal{L}}_i}\{x_i \in P_i^{(l)} | K_i^{(l)} x_i \in \mathcal{U}_i\}$, terminating in the robustly positive invariant set $\mathcal{X}_{\mathrm{T},i}$, with each set in the sequence being a contraction of the previous one. See [13] for construction of $\mathcal{X}_{0,i}$.

In this appendix, a modified procedure with respect to that in [13] is presented for which the resulting robustly positive invariant sets are robust only to coupling for neighboring subsystems within their own local terminal sets, as by Assumption 4. A formal proof of the positive invariance, missing in [13], is also provided.

Define the matrix $A_{\mathrm{cl},i}^{(l)} = A_i^{(l)} + B_i^{(l)} K_i^{(l)}$. Define the set operation

$$\mathcal{Q}_i^{(l)}(\mathcal{X}, \mathcal{W}) = \{x_i \in \mathbb{R}^{n_i} | A_{\mathrm{cl},i}^{(l)} x_i + w \in \mathcal{X} \quad \forall w \in \mathcal{W}\} = (A_{\mathrm{cl},i}^{(l)})^{-1}(\mathcal{X} \ominus \mathcal{W}) \quad (46)$$

i.e., a one-step controllable set, robust to disturbances in the set $\mathcal{W}$. Algorithm 6 outlines the procedure for computing $\mathcal{X}_{\mathrm{T},i}$ for $i \in \mathcal{M}$.

Assumption 7 (See [13]): For all $i \in \mathcal{M}$ there exists $\tau \in \mathbb{N}^+$ and $\tau < \infty$ such that $\mathcal{X}_{\tau,i} = \mathcal{X}_{\tau-1,i}$ and $\mathcal{X}_{\tau,i} \neq \emptyset$.

Theorem 3: Under Assumption 7 the output sets from Algorithm 6 are robustly positive invariant as in Assumption 4.

Proof: As at each iteration τ of Algorithm 6 the set $\mathcal{X}_{\tau,i}$ shrinks, then for all $i \in \mathcal{M}$, if $x_i \in \mathcal{X}_{\mathrm{T},i}$ and $x_j \in \mathcal{X}_{\mathrm{T},j}$ for all $j \in \mathcal{N}_i$, then $x_i \in \mathcal{X}_{\tau,i}$ and $x_j \in \mathcal{X}_{\tau-1,i}$ for all $j \in \mathcal{N}_i$ and for all $\tau \geq 1$. Then, by the definitions of $\mathcal{Q}_i^{(l)}$ and $\mathcal{W}_{\tau-1,i}$, $A_{\mathrm{cl},i}^{(l)} x_i + \sum_{j\in\mathcal{N}_i} A_{ij} x_j \in \mathcal{X}_{\tau-1,i}$ for all $l \in \hat{\mathcal{L}}_i$ and all $\tau \geq 2$.

Algorithm 6: Computation of $\mathcal{X}'_{\mathrm{T},i}s$.

Inputs: Initial sets $\mathcal{X}_{0,i}\ \forall i \in \mathcal{M}$
Initialize: $\mathcal{X}_{1,i} \leftarrow \emptyset\ \forall i \in \mathcal{M}$ and $\tau \leftarrow 1$
while $\mathcal{X}_{\tau,i} \neq \mathcal{X}_{\tau-1,i}$ for some $i \in \mathcal{M}$
  for $i \in \mathcal{M}$ **do**
    $\mathcal{W}_{\tau-1,i} \leftarrow \bigoplus_{j\in\mathcal{N}_i} A_{ij}\mathcal{X}_{j,\tau-1}\ \forall j \in \mathcal{N}_i$
    $\mathcal{X}^{(l)}_{\tau,i} \leftarrow \mathcal{Q}^{(l)}_i(\mathcal{X}_{\tau-1,i}, \mathcal{W}_{\tau-1,i}) \cap \mathcal{X}_{\tau-1,i}\ \forall l \in \hat{\mathcal{L}}_i$
    $\mathcal{X}_{\tau,i} \leftarrow \bigcap_{l\in\hat{\mathcal{L}}_i} \mathcal{X}^{(l)}_{\tau,i}$
  end for
  $\tau \leftarrow \tau + 1$
end while
$\mathcal{X}_{\mathrm{T},i} \leftarrow \mathcal{X}_{\tau-1,i}\ \forall i \in \mathcal{M}$
Outputs: $\mathcal{X}_{\mathrm{T},i}\ \forall i \in \mathcal{M}$

Then, $A^{(l)}_{\mathrm{cl},i}x_i + \sum_{j\in\mathcal{N}_i} A_{ij}x_j \in \mathcal{X}_{\mathrm{T},i}$ for all $l \in \hat{\mathcal{L}}_i$. Hence, $\mathcal{X}_{\mathrm{T},i}$ is robustly positive invariant, robust to the coupling of neighboring subsystems within their respective terminal sets, under arbitrary switching between the PWA regions $l \in \hat{\mathcal{L}}_i$. It follows directly that $\mathcal{X}_{\mathrm{T},i}$ is robustly positive invariant for the PWA sub-system under the true switching. □

Appendix B
Computation of $V_{f,i}$

Given the structures of ℓ_i and $V_{\mathrm{T},i}$ (26) can be written as

$$\sum_{i\in\mathcal{M}} (\star)^\top \Phi_i \left((A^{(l_i)}_i + B^{(l_i)}_i K^{(l_i)}_i)x_i + \sum_{j\in\mathcal{N}_i} A^{(l_i)}_{ij}x_j \right) - x_i^\top \Phi_i x_i + x_i^\top Q_i x_i + (\star)^\top R_i (K^{(l_i)}_i x_i) + \sum_{j\in\mathcal{N}_i} x_j^\top Q_{ij} x_j \leq 0, x_i \in P^{(l_i)}_i \tag{47}$$

for all $x_i \in \mathcal{X}_{\mathrm{T},i}, l_i \in \hat{\mathcal{L}}_i, i \in \mathcal{M}$, recalling that $c^{(l_i)}_i = 0$ for $l_i \in \hat{\mathcal{L}}_i$. Finding Φ_i's such that (47) is satisfied can be achieved by considering the system from a global view, ensuring the inequality holds for every combination of PWA regions of the subsystems. This is equivalent to considering the global system as a single large PWA system with $\prod_{i\in\mathcal{M}} L_i$ regions and using the approach presented in [21] for single PWA systems. Condition (47) can be reformulated as the set of LMIs

$$A^\top_{\mathrm{cl}}(l)\Phi A_{\mathrm{cl}}(l) - \Phi + Q + K(l)RK(l) \prec 0 \quad \forall l = (l_1, \ldots, l_M) \in \hat{\mathcal{L}}_1 \times \cdots \times \hat{\mathcal{L}}_M \tag{48}$$

where $\Phi = \mathrm{blck}(\Phi_1, \ldots, \Phi_M)$, $Q = \mathrm{blck}(Q_1 + \sum_{j|1\in\mathcal{N}_j} Q_{j1}, \ldots, Q_M + \sum_{j|M\in\mathcal{N}_j} Q_{jM})$, $R = \mathrm{blck}(R_1, \ldots, R_M)$, $K(l) = \mathrm{blck}(K^{(l_1)}_1, \ldots, K^{(M)}_M)$, and

$$A_{\mathrm{cl}}(l) = \begin{bmatrix} A^{(l_1)}_{\mathrm{cl},1} & A^{(l_1)}_{12} & \cdots & A^{(l_1)}_{1M} \\ A^{(l_2)}_{21} & \ddots & & \vdots \\ \vdots & & \ddots & \\ A^{(l_M)}_{M1} & & & A^{(l_M)}_{\mathrm{cl},M} \end{bmatrix} \tag{49}$$

with $A^{(l_i)}_{ij} = 0$ if $j \notin \mathcal{N}_i$. Evidently, (48) requires centralized computation, and may become intractable for large M or $|\hat{\mathcal{L}}_i|$. We leave, however, distributed and consequently less computationally demanding solutions to future work. Finally, (48) is linear in the matrix Φ as the linear controllers $K^{(l)}_i$ have been assumed to be computed a priori as in [13]. The linear controllers could be computed simultaneously using the same expression, leveraging the Schur complement and a change of variables to maintain an LMI expression [21].

Appendix C
Suboptimality of Algorithm 3

To investigate the suboptimality of Algorithm 3, the MPC optimization problem $\mathcal{P}_\mathrm{d}$ is solved for 1000 randomly generated two-subsystem PWA networks and states. Each subsystem has two states, one input, and four PWA regions. All values defining the local state-space matrices take random values between $\begin{bmatrix}-1.5 & 1.5\end{bmatrix}$ (stable and unstable systems), while the coupling matrices take random values between $\begin{bmatrix}-0.1 & 0.1\end{bmatrix}$. The four PWA regions are the four quadrants of the Cartesian plane. The local stage costs are $\ell_i = x_i^\top x_i + u_i^\top u_i$, and the local constraints are $|u_i| \leq 1$ and $\|x_i\|_\infty \leq 10$. No terminal costs or constraints are considered. Problem $\mathcal{P}_\mathrm{d}$ is solved by Algorithm 3 for states x randomly selected with values in the range $\begin{bmatrix}-10 & 10\end{bmatrix}$, with $\mathbf{u}$ the resulting global control sequence. To avoid the manual tuning of ρ, T_{ADMM}, and T_{cut} for each random system, T_{cut} is set to 100 and the algorithm is run until the residual is less than 0.01. Control sequences of all zeros are used as weakly feasible initial guesses. Problem $\mathcal{P}_\mathrm{d}$ is also reformulated as an MIQP and solved to optimality with Gurobi [27], with $\mathbf{u}^*$ the optimal global control sequence. Over all randomized networks and states the suboptimality

$$\tilde{V} = 100 \cdot \frac{V(x,\mathbf{u}) - V(x,\mathbf{u}^*)}{V(x,\mathbf{u}^*)} \tag{50}$$

has median 0, mean 16.0, standard deviation 120.7, maximum 1664.2, and minimum 0. This demonstrates that, while there exists cases in which arbitrarily bad local minima are found, these are outliers, with Algorithm 3 in general finding solutions with low suboptimality. As demonstrated and discussed in Section V, the suboptimality can be reduced using a multistart approach, and, when Algorithm 5 is deployed, stability is guaranteed.

Acknowledgment

The authors would like to thank the A. Ma, D. Li, and Y. Li, for helpfully providing the code for their approach.

References

[1] F. Lamnabhi-Lagarrigue et al., "Systems & control for the future of humanity, research agenda: Current and future roles, impact and grand challenges," *Annu. Rev. Control*, vol. 43, pp. 1–64, 2017.
[2] X. Luan, B. De Schutter, L. Meng, and F. Corman, "Decomposition and distributed optimization of real-time traffic management for large-scale railway networks," *Transp. Res. Part B: Methodological*, vol. 141, pp. 72–97, 2020.

[3] P. R. C. Mendes, J. M. Maestre, C. Bordons, and J. E. Normey-Rico, "A practical approach for hybrid distributed MPC," *J. Process Control*, vol. 55, pp. 30–41, 2017.
[4] H. Ekeren, R. Negenborn, P. van Overloop, and B. De Schutter, "Time-instant optimization for hybrid model predictive control of the rhine–meuse delta," *J. Hydroinformatics*, vol. 15, no. 2, pp. 271–292, 2013.
[5] E. Sontag, "Nonlinear regulation: The piecewise linear approach," *IEEE Trans. Autom. Control*, vol. 26, no. 2, pp. 346–358, Apr. 1981.
[6] W. P. M. H. Heemels, B. De Schutter, and A. Bemporad, "Equivalence of hybrid dynamical models," *Automatica*, vol. 37, no. 7, pp. 1085–1091, 2001.
[7] J. M. Maestre and R. R. Negenborn, Eds., *Distributed Model Predictive Control Made Easy*. Berlin, Germany: Springer, 2014.
[8] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.
[9] S. Boyd, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2010.
[10] A. Bemporad and M. Morari, "Control of systems integrating logic, dynamics, and constraints," *Automatica*, vol. 35, no. 3, pp. 407–427, 1999.
[11] D. Großand and O. Stursberg, "Distributed predictive control for a class of hybrid systems with event-based communication," *IFAC Proc. Volumes*, vol. 46, no. 27, pp. 383–388, 2013.
[12] Y. Kuwata, A. Richards, T. Schouwenaars, and J. P. How, "Distributed robust receding horizon control for multivehicle guidance," *IEEE Trans. Control Syst. Technol.*, vol. 15, no. 4, pp. 627–641, Jul. 2007.
[13] A. Ma, D. Li, and Y. Xi, "Distributed MPC based on robustly controllable sets for PWA systems," *Automatica*, vol. 154, 2023, Art. no. 111078.
[14] A. Bemporad, *Modeling, Control, and Reachability Analysis of Discrete-Time Hybrid Systems*. Sienna, Italy: Univ. Sienna, 2003.
[15] T. H. Summers and J. Lygeros, "Distributed model predictive consensus via the alternating direction method of multipliers," in *Proc. 50th Annu. Allerton Conf. Commun., Control, Comput.*, 2012, pp. 79–84.
[16] P. Giselsson and A. Rantzer, "On feasibility, stability and performance in distributed model predictive control," *IEEE Trans. Autom. Control*, vol. 59, no. 4, pp. 1031–1036, Apr. 2014.
[17] J. Köhler, M. A. Müller, and F. Allgöwer, "Distributed model predictive controlrecursive feasibility under inexact dual optimization," *Automatica*, vol. 102, pp. 1–9, 2019.
[18] D. Q. Mayne and S. Rakovic, "Model predictive control of constrained piecewise affine discrete-time systems," *Int. J. Robust Nonlinear Control*, vol. 13, no. 3–4, pp. 261–279, 2003.
[19] R. Rostami and D. Görges, "An ADMM-based algorithm for stabilizing distributed model predictive control without terminal cost and constraint," *Eur. J. Control*, vol. 73, 2023, Art. no. 100881.
[20] P. O. M. Scokaert, D. Q. Mayne, and J. B. Rawlings, "Suboptimal model predictive control (feasibility implies stability)," *IEEE Trans. Autom. Control*, vol. 44, no. 3, pp. 648–654, Mar. 1999.
[21] M. Lazar, W. P. M. H. Heemels, S. Weiland, and A. Bemporad, "Stabilizing model predictive control of hybrid systems," *IEEE Trans. Autom. Control*, vol. 51, no. 11, pp. 1813–1818, Nov. 2006.
[22] C. Conte, C. N. Jones, M. Morari, and M. N. Zeilinger, "Distributed synthesis and stability of cooperative distributed model predictive control for linear systems," *Automatica*, vol. 69, pp. 117–125, 2016.
[23] A. Richards and J. P. How, "Robust distributed model predictive control," *Int. J. Control*, vol. 80, no. 9, pp. 1517–1531, 2007.
[24] F. Asadi and A. Richards, "Scalable distributed model predictive control for constrained systems," *Automatica*, vol. 93, pp. 407–414, 2018.
[25] H. Lin and P. J. Antsaklis, "Stability and stabilizability of switched linear systems: A survey of recent results," *IEEE Trans. Autom. Control*, vol. 54, no. 2, pp. 308–322, Feb. 2009.
[26] S. Mallick, A. Dabiri, and B. De Schutter, "A comparison benchmark for distributed hybrid MPC control methods: Distributed vehicle platooning," 2024, *arXiv:2401.09878*.
[27] Gurobi Optimization, LLC, "Gurobi optimizer reference manual," 2023 [Online]. Available: https://www.gurobi.com
[28] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "OSQP: An operator splitting solver for quadratic programs," *Math. Program. Computation*, vol. 12, no. 4, pp. 637–672, 2020, [Online]. Available: https://doi.org/10.1007/s12532-020-00179-2
[29] H. Ferreau, C. Kirches, A. Potschka, H. Bock, and M. Diehl, "qpOASES: A parametric active-set algorithm for quadratic programming," *Math. Program. Computation*, vol. 6, no. 4, pp. 327–363, 2014.
[30] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi—A software framework for nonlinear optimization and optimal control," *Math. Program. Computation*, vol. 11, no. 1, pp. 1–36, 2019.
[31] I. I. Cplex, "V12. 1: User's manual for CPLEX," *Int. Bus. Mach. Corporation*, vol. 46, no. 53, 2009, Art. no. 157.
[32] M. ApS, "Mosek optimization toolbox for matlab," *User's Guide Reference Manual, Version*, vol. 4, no. 1, 2019 .

Samuel Mallick received the B.Sc. and M.Sc. degrees in mechatronics engineering from The University of Melbourne, Parkville, VIC, Australia, in 2020 and 2022, respectively. He is currently working toward the Ph.D. degree in systems and control with the Delft Center for Systems and Control, Delft University of Technology, The Netherlands.

His research interests include model predictive control, reinforcement learning, and distributed control of large-scale and hybrid systems.

Azita Dabiri received the Ph.D. degree in systems and control from the Automatic Control Group, Chalmers University of Technology, Gothenburg, Sweden, in 2016.

From 2017 to 2019, she was a Postdoctoral Researcher with the Department of Transport and Planning, Delft University of Technology, The Netherlands. In 2019, she received an ERCIM Fellowship and also a Marie Curie Individual Fellowship, which allowed her to perform research at the Norwegian University of Technology (NTNU), as a Postdoctoral Researcher, from 2019 to 2020, before joining the Delft Center for Systems and Control, Delft University of Technology, as an Assistant Professor. Her research interests include areas of integration of model-based and learning-based control and its applications in transportation networks.

Bart De Schutter (Fellow, IEEE) received the Ph.D. degree (*summa cum laude*) in applied sciences from KU Leuven, Leuven, Belgium, in 1996.

He is currently a Full Professor and Head of Department with the Delft Center for Systems and Control, Delft University of Technology, The Netherlands. His research interests include multilevel and multiagent control, model predictive control, learning-based control, and control of hybrid systems, with applications in intelligent transportation systems and smart energy systems.

Dr. De Schutter is a Senior Editor of IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS.