Coalitional Distributed Model Predictive Control Strategy for Vehicle Platooning Applications

Anca Maxim and Constantin-Florin Caruntu*

Department of Automatic Control and Applied Informatics, “Gheorghe Asachi” Technical University of Iasi, 700050 Iasi, Romania; anca.maxim@ac.tuiasi.ro
* Correspondence: caruntuc@ac.tuiasi.ro

Abstract: This work aims at developing and testing a novel Coalitional Distributed Model Predictive Control (C-DMPC) strategy suitable for vehicle platooning applications. The stability of the algorithm is ensured via the terminal constraint region formulation, with robust positively invariant sets. To ensure a greater flexibility, in the initialization part of the method, an invariant table set is created containing several invariant sets computed for different constraints values. The algorithm was tested in simulation, using both homogeneous and heterogeneous initial conditions for a platoon with four homogeneous vehicles, using a predecessor-following, uni-directionally communication topology. The simulation results show that the coalitions between vehicles are formed in the beginning of the experiment, when the local feasibility of each vehicle is lost. These findings successfully prove the usefulness of the proposed coalitional DMPC method in a vehicle platooning application, and illustrate the robustness of the algorithm, when tested in different initial conditions.

Keywords: coalitional model predictive control; distributed model predictive control; robust positively invariant sets; closed-loop stability; vehicle platooning

1. Introduction

One of the recent interests in the control community is focused on developing advanced control techniques in the framework of autonomous vehicles. Thus, considerable efforts were made to increase the efficiency and to improve the collaboration through the connectivity between multiple autonomous vehicles, using well established control strategies, such as Model Predictive Control (MPC) or Distributed Model Predictive Control (DMPC) [1]. Among these strategies, vehicle platooning of connected and autonomous vehicles has increased potential [2]. The idea behind this methodology is to control a group of vehicles, such that the entire ensemble travels with a constant velocity, while maintaining a desired, safe inter-vehicle distance [3]. One of the most common vehicle platooning frameworks uses the predecessor-following communication topology. Each platoon vehicle receives information only from its preceding vehicle, via a vehicle-to-vehicle (V2V) communication technology [4].

For vehicle platooning applications, several control approaches were proposed in the state-of-the-art literature, see the review paper [5]. Thus, a distributed adaptive robust $H_{\infty}$ control strategy for vehicle platooning is given in [6]. Here, an adaptive robust $H_{\infty}$ controller is compared in simulation with a robust $H_{\infty}$ controller in a seven vehicles platoon application. The platoon stability is formulated using a Linear Matrix Inequality (LMI) condition derived from the Lyapunov stability theory. Another approach, based on stochastic dynamic programming is provided [7]. The idea is to describe the coordinated platooning problem of autonomous vehicles at highway junctions as a Markov decision process. In [8], a decentralized multi-agent game-theoretical strategy is proposed, in which individual autonomous vehicles form platoons using either one-to-one or many-to-many coordination. Furthermore, two game-theoretical models are proposed, with complete or incomplete information exchanged between agents. The method is validated...
in simulation, in various tests in which, seven to twenty-five vehicles need to coordinate and form different platoons of maximum size of seven vehicles. In [9], a dynamic event-triggered control and communication strategy is presented. The algorithm is suitable to reduce the utilization of the wireless communication network and ensures a $L_2$ string-stable platoon. The methodology that was tested in the simulation is a three vehicle platoon. Furthermore, in [10], a DMPC strategy for vehicle platooning with different types of communication delays is proposed. In [11], a min-max DMPC strategy for vehicle platoons with communication delays is presented. The algorithm provides input-to-state stability margins for the platoon analysis. Each vehicle from the platoon is modelled using a point-mass model, which is the simplest model available, that ignores the vehicle dynamics. In [12], a non-iterative DMPC strategy for flexible vehicle platooning is given. The algorithm implements a vehicle partitioning procedure, suitable when an individual vehicle wants to join an existing platoon. The methodology assumes the platoon partition into different disjoint vehicle clusters, connected through connecting vehicle clusters, which act as common neighbours between the consecutive disjoint clusters. The algorithm is tested on a point-mass vehicle model. In [13], a DMPC algorithm for vehicle platoons with switching communication topologies is presented. Starting from the basic predecessor-leader communication topology, the algorithm switches to another topology, when a communication failure occurs. Thus, if the leader vehicle can not send information, the first follower becomes the new leader of the platoon, whereas when a follower vehicle fails to send information, this is automatically excluded from the platoon communication. The latter will act as an independent vehicle, which physically is still in the platoon, but must rely on its own sensors to maintain a constant inter-vehicle distance with its successor. The string stability analysis of the platoon is formulated in $L_2$-norm of the spacing error. The platoon is also modelled with a point-mass vehicle model.

The DMPC method emerged from the mature MPC field and is suitable for complex systems that can be decomposed into smaller interconnected sub-systems. To relieve the computational burden of the methodology, each sub-system is locally controlled by an agent, which solves a smaller optimization problem depending on both local measured information acquired by the sensors and data exchanged within a neighbourhood. This means that all agents corresponding to sub-systems, which are coupled (dynamically or through the constraints or the cost functions), need to exchange relevant information, to ensure local feasibility for the optimization problems [14,15].

Another recent development within this methodology is the Coalitional Control, with characteristic features from cooperative game-theory [16]. The idea is to consider that the interconnected sub-systems have a dynamic communication architecture, which can enable or disable the existing communication links, according to the specific overall control objectives [17].

Following the idea of a flexible communication architecture between coupled sub-systems, a Coalitional Distributed Model Predictive Control (C-DMPC) strategy for a cyber-physical multi-agent application was proposed in [18]. The key feature for this methodology is to use the advantages provided by the DMPC algorithm by creating an adjustable controller framework. The architecture starts from a non-cooperative DMPC structure (i.e., the agents solve local optimization problems taking into account their respective interactions), and evolves into coalitional DMPC, e.g., groups of agents merge in a single entity or cooperative group (hereafter called a coalition) and commonly solve a global problem with respect to the coalition objective [18,19].

Regardless if the DMPC or the C-DMPC framework is considered, since both are derived from the MPC strategy, they inherit one key feature, namely the accommodation of constraints directly during the optimal-design procedure [20,21]. From the dynamical stability point of view, one method to ensure the stability of the MPC algorithm (and implicitly the C-DMPC strategy) is formulating it using the terminal region constraint framework, which can be defined as an invariant set [22,23]. There is a high interest in the literature to develop reliable methods for computing invariant sets for uncertain
systems, in which the disturbance acts as an uncertainty in the nominal model [24,25]. Several approaches to calculate robust control invariant sets can be mentioned, such as those based on: graph theory and algorithms [26], linear matrix inequalities [27], Hamilton–Jacobi reachability framework [28] or dynamic programming [29]. In [30], the boundary of the maximal robust positively invariant set is studied by adapting the theory of barriers formulation. The latter is then used in [31], to perform a transient stability analysis for power systems. In [32], an analysis regarding the scalability of robust positively invariant sets with respect to the disturbance set is provided. In [33], robust positive invariant sets are computed using sequence-based techniques and tested on a switched system with average dwell time.

In this work, the focus is on developing and testing a novel C-DMPC methodology. The algorithm is designed for state-coupled sub-systems, in which the interaction between consecutive modules is considered an uncertainty in the states, rather than the input-coupled C-DMPC formulation from [18]. Emphasis is given to the computational procedure used to ensure the closed-loop stability of the algorithm, thus providing the technical details regarding the computation of the maximal robust positively invariant sets, instead of giving the basic pseudo-code algorithm [18]. Hence, to ensure several available options for the terminal region constraint set, a predefined table is computed in the initialization stage of the C-DMPC algorithm. Each cell in this table is a maximal robust positively invariant set, computed for a distinct parametrization of the input, state, and uncertainty constraint values. Thus, the invariant sets table is tailored specifically for a realistic vehicle platooning application, rather than the academic model of a cyber–physical multi-agent system from [18]. A vehicle platoon application is envisioned, consisting of four vehicles, dynamically coupled through the states. The model takes into account the dynamics of the vehicle, as opposed to the DMPC algorithms provided in [11–13], which were tested using a point-mass vehicle model. Each vehicle is locally controlled by an agent connected through an uni-directional channel, in a predecessor-following communication topology. The interaction between sub-systems is considered as state uncertainty for the nominal local model. This means that between consecutive vehicles, each agent in-charge needs to receive information from its predecessor and transmit data to its successor. Another key feature within the C-DMPC algorithm is the amount of information exchanged between agents, which is minimized by defining the uncertainties as boxes of fixed size, and transmitting only the bounding limits. Based on this uncertainty value, the feasibility of each local optimization problem is checked, and in case of infeasibility, a coalition between neighbouring agents is formed. The C-DMPC methodology is formulated to ensure zero-state minimization for the local and coalitional optimization problems, which is described in a more straightforward manner than in [18], where an integral state is introduced to ensure zero offset reference tracking. The robustness of the algorithm is analysed in various simulation tests, using a homogeneous platoon, with different initial conditions, both homogeneous and heterogeneous, under the same optimization parameters and constraints. The string stability of the platoon is analysed a posteriori, i.e., by verifying that the condition from [34] is satisfied on the obtained results, rather than imposing it as a constraint in the optimization problem. Moreover, the string stability of the platoon is investigated taking into account the nature of the proposed C-DMPC control framework. When a coalition is formed, the agents inside are treated as one entity with respect to the remaining agents outside of the coalition, which is different from the string stability analysis provided in a DMPC context in [35].

Taking all the above mentioned into considerations, the main contributions for this work, when compared to our previous papers and state-of-the-art literature are as follows:
1. A novel C-DMPC methodology, suitable for a state-coupled vehicle platooning application is developed.
2. The control performances of the algorithm were analysed for simulation tests using a realistic vehicle platoon model from the literature.
3. Both the robustness of the algorithm and the string stability of the platoon were investigated.

The remainder of this paper is structured as follows: Section 2 introduces the problem formulation, while the details regarding the computation of the invariant sets table are presented in Section 3. A summary of the C-DMPC method is provided in Section 4, followed by the simulation results and discussion given in Section 5. The conclusions and future work ideas are presented in Section 6.

Notation: \( \mathbb{N} \) denotes the set of natural numbers; \( \mathbb{Z} \) denotes the set of integer numbers; \( \mathbb{Z}_+ \) denotes the set of positive integer numbers; \( \mathbb{R} \) denotes the set of real numbers; \( \mathcal{X} \) denotes a closed convex polytope in H-representation, written as a matrix inequality \( \mathcal{X} = \{ x \in \mathbb{R} : Ax \leq b \} \); \( \| \cdot \|_1 \) denotes the 1-norm; and \( \| \cdot \|_\infty \) denotes the \( \infty \)-norm.

2. Problem Formulation

Let us consider a multi-agent system consisting of \( N \) interconnected pairs \( (S_i, A_i) \), \( \forall i \in \mathcal{N} \), where \( \mathcal{N} \) denotes the set \{1, \ldots, \( N \)\} \( \subseteq \mathbb{N} \), \( S_i \) denotes the sub-system and \( A_i \) denotes the agent (or local controller) in charge of \( S_i \). In the remaining of this paper, we consider that all sub-systems \( S_i \) have a nominal model with state uncertainty, described as follows:

\[
\begin{align*}
    x_i(k+1) &= A_{ij}x_i(k) + B_{ij}u_i(k) + w_i(k) \\
    w_i(k) &= A_{ij-1}x_i(k)
\end{align*}
\]

where \( x_i \in \mathbb{R}^{n_i} \) is the state, \( u_i \in \mathbb{R}^{m_i} \) is the input, \( w_i \in \mathbb{R}^{p_i} \) is the state uncertainty and \( y_i \in \mathbb{R}^{q_i} \) is the output. \( A_{ij}, B_{ij}, A_{ij-1} \) and \( C_i \) are matrices with adequate dimensions. \( n_i, m_i, p_i \) and \( q_i \) are the number of states, inputs, state uncertainties and outputs, respectively, and \( k \in \mathbb{Z}_+ \) is the discrete-time instant.

It is noteworthy to mention that in (1) the nominal model contains a state uncertainty \( w_i \), defined based on the dynamical coupling of the sub-systems. Since all the sub-systems \( S_i, \forall i \in \mathcal{N} \), are unidirectionally interconnected, the relevant information is received from each predecessor denoted with index \( i-1 \).

Consider linear inequality constraints for the states, inputs and uncertainties defined with:

\[
\begin{align*}
    x_i \in \mathcal{X}_i, & \quad u_i \in \mathcal{U}_i, & \quad w_i \in \mathcal{W}_i, & \quad \forall i \in \mathcal{N},
\end{align*}
\]

where \( \mathcal{X}_i, \mathcal{U}_i \) and \( \mathcal{W}_i \) denote the polytopes obtained from the state, input and uncertainty constraints, respectively.

Further on, the local optimization problem solved by each agent \( A_i, \forall i \in \mathcal{N} \), is described in the min-max framework, by minimizing the control input for the maximum level of uncertainty, which will be received from its predecessor agent. Thus, the non-cooperative DMPC cost function solved at each sampling instant is as follows:

\[
J_i(x_i^0) = \min_{u_i^0, \ldots, u_i^{N_p-1}} \max_{x_i^{\text{max}}, x_i^{\text{min}}} \max_{u_i^0, \ldots, u_i^{N_p-1}} \sum_{j=0}^{N_i-1} \| R_{ij} x_i^{\text{max}} \|_1 + \| R_{ij} x_i^{\text{min}} \|_1 + \| x_i^{N_p} \|_1
\]

s. t. (1) \hspace{2cm} (3)

\[
\begin{align*}
    x_i^{\text{min}} &\leq x_i^l \leq x_i^{\text{max}}, & \quad l = 1, \ldots, N_p - 1 \\
    x_i^{N_p} &\in \Omega_i \\
    u_i^{\text{min}} &\leq u_i^l \leq u_i^{\text{max}}, & \quad l = 1, \ldots, N_p - 1 \\
    w_i^l &\leq w_i^{\text{max}}
\end{align*}
\]
with the following notations for sub-system $S_i$: $x_i^l = x_i(k+l|k)$ is the output predictor computed at sample time $k$, for time $k+l$, within the prediction horizon $N_p, l \in \{1, \ldots, N_p\}$; the output trajectory is obtained recursively using model (1), the state measurement $x_i^0 = x_i(k)$ at time $k$, the input trajectory $u_i^l = \{u_i^0, \ldots, u_i^{N_p-1}\}$ and the uncertainty trajectory $w_i^l = \{w_i^0, \ldots, w_i^{N_p-1}\}$, i.e., $W_i = A_{i(j-1)}X_{i-1}$ received from agent $A_{i-1}$; $x_i^{N_p}$ is the value for the state predictor, for time $k+N_p$; $x_i^{\text{max}}$, $u_i^{\text{max}}$, $w_i^{\text{max}}$ are the maximum limits for the state, input and the uncertainty sequences, respectively; $x_i^{\text{min}}$, $u_i^{\text{min}}$ are the minimum limits for the state and input sequences, respectively; $R_{u_i} \in \mathbb{R}^{m_i \times m_i}$ and $R_{w_i} \in \mathbb{R}^{n_i \times n_i}$ are the weight matrices for the input trajectory, and the optimization variable $\hat{x}^{\text{max}}$ that acts as a self-imposed limit for the state sequence. After the local optimization problem is solved, the self-imposed value will be transmitted to the neighbour, which will use it as a boundary for the uncertainty variable. In this manner, each local agent is assured that the actual uncertainty trajectory (which is not communicated) is bounded at the received value.

The set $\Omega_i$ is a maximal robust positively invariant set for the output terminal constraint, and its computation will be detailed in the following.

3. Computation of the Maximal Robust Positively Invariant Set

In this section, the computation of the maximal robust positively invariant set $\Omega_i$, $\forall i \in \mathcal{N}$, which will be used as a terminal output region $x_i^{N_p} \in \Omega_i$ is described. The procedure was initially presented in [19] and particularized in [18] for the sub-system model with an integral additional state.

Hereafter, a short summary is provided, followed by details regarding the computation of the invariant sets.

Let us consider the following assumptions:

- Each sub-system $S_i, \forall i \in \mathcal{N}$, is modelled by (1) and subject to constraints (2);
- For the proposed coalitional framework, in the case of communication network failure, the information exchange between neighbouring agents stops. To ensure the feasibility of the control architecture, the invariant sets are tailored for the nominal model (in which the uncertainty $w_i$ is zero);
- A local linear feedback $u_i = K_i x_i$ that is computed to guarantee that the stability of the closed loop is a-priory determined. In this paper, the state feedback matrix $K_i$ is computed with two methods: (i) using Ackermann’s formula [36] and (ii) through the minimization of a linear-quadratic cost function [37].

For the sub-system (1) subject to the constraints (2), the set $\Omega_i$ is maximal robust positively invariant if [18,19,38]:

$$x_i \in \Omega_i \rightarrow (A_{i(j)} + B_{i(j)}K_i)x_i + w_i \in \Omega_i,$$

$$K_i x_i \in U_i, \Omega_i \subseteq X_i, \forall w_i \in W_i \tag{4}$$

Hereafter, the pseudo-code algorithm provided in [18], which describes how to compute the predefined invariant set table is summarized and will be further detailed.

As previously mentioned, during the initialization phase, each agent $A_i, \forall i \in \mathcal{N}$, calculates a table of invariant sets $\Omega_i$ by applying Algorithm 1.
Algorithm 1

For $\alpha = \alpha_{\text{max}} : -\text{step} \alpha : \alpha_{\text{min}}$
For $\beta = \beta_{\text{min}} : \text{step} \beta : \beta_{\text{max}}$

1. Compute the inequality constraints:
   \[ A_u u \leq \alpha b_u; \quad A_w w \leq \beta b_w; \quad A_x x \leq b_x \]

2. Compute the robust positively invariant set:
   \[ \Omega(A, B, K, A_u, b_u, A_w, b_w, A_x, b_x) \]

3. Save the information $\alpha, \beta, \Omega$

end
end

Further on, several computational details regarding the second step from Algorithm 1 (i.e., how to compute the robust positively invariant set) are illustrated. For simplicity, the indices are omitted:

1. Compute the closed loop system matrix using the sub-systems model matrices $A$, $B$, and control feedback gain $K$;
2. Re-write the input inequality constraints from Algorithm 1, step 1, i.e., described in matrix form using matrix $A_u$ and vector $b_u$ with adequate dimensions, in the state variable $x$ using $u = Kx$;
3. Aggregate state and input constraints from step 2 in a single polytope;
4. Compute the uncertainty polytope using the uncertainty constraints from Algorithm 1, step 1;
5. Iteratively, check the invariance order by executing the following:
   (a) Subtract the uncertainty polytope from the aggregated polytope from step 3;
   (b) Compute the polytope that, from the previous iteration, using the dynamics of the control loop, arrived in the current region (i.e., defined as step 5(a));
   (c) Intersect the two polytopes obtained at steps 5(a) and 5(b);
   (d) Compare the result of the polytopes intersection—from step 5(c) with the aggregated constraint polytope from step 3;
   (e) If the invariant set has changed
       then: Update the constraint polytope (with the value computed from step 5(c)) and continue the iterations;
       else: Save the invariant set and end the iterations.

It is worth mentioning that the maximal robust positively invariant set is computed recursively (as shown in step 5) for a finite number of iterations. If by the end of those, the desired region is not found, then the algorithm will return an empty set, meaning that for the particular combination of constraints values, there is no available invariant set (i.e., the corresponding cell in the table is empty).

4. Coalitional Distributed Model Predictive Control (C-DMPC) Algorithm

In this paper, a novel C-DMPC methodology was designed for sub-systems coupled through the states, which must exchange the state-uncertainty maximum value. Moreover, the algorithm which was firstly introduced in [18] for cyber–physical systems, was tailored in this work for a vehicle platooning application, and customized to ensure zero-state minimization. Hereafter, a concise summary is provided:

- At each sampling time, each agent solves a default robust min-max non-cooperative DMPC problem with state-uncertainty. If the local solution is infeasible, due to the uncertainty value received from its neighbour, then the coalition procedure between the agents is activated ad hoc without a supervisor level.
Inside a coalition $C$, the local models of each involved sub-systems are merged, obtaining the following coalition model (written for simplicity without sub-script indices):

$$
\begin{align*}
    x_C(k+1) &= A_C x_C(k) + B_C u_C(k) + w_C(k) \\
    w_C(k) &= \sum_{j \in N_C} A_{C,j} x_j(k)
\end{align*}
$$

(5)

where $x_C$, $u_C$, $w_C$ are the aggregated state, input and uncertainty vectors of the coalition $C$ (e.g., $x_C = [x_i]_{i \in C}$). Moreover, the matrices $A_C$, $B_C$, $A_{C,j}$ and $C_C$ are computed according to the aggregation. The set $N_C$ contains the neighbouring agent for $C$ neighbour, i.e., the predecessor sub-system for the sub-systems included in the coalition. For example, a coalition between agents $A_3$ and $A_4$ remains state-coupled with $A_2$, which is outside the coalition and sends information to $A_3$. It follows that, the constraint sets for a given coalition $C$ are obtained as the union of the constraints sets (2) corresponding to each agent $A_i, i \in C$:

$$
\begin{align*}
    x_C \in X_C = \prod_{i \in C} X_i, \quad u_C \in U_C = \prod_{i \in C} U_i, \quad w_C \in W_C = \prod_{i \in C} W_i.
\end{align*}
$$

(6)

where $X_C$, $U_C$ and $W_C$ are the state, input and uncertainty constraints polytopes for coalition $C$, respectively.

A coalition once formed is active only for a single sampling period and can gradually increase its size. Hence, the feasibility for the coalition optimization problem is verified. If the solution is infeasible, this means that the received uncertainty value is too large, and an agent from the neighbourhood must be incorporated in the coalition, to decrease the uncertainty value. For example, if the coalition between $A_3$ and $A_4$ has an infeasible solution, then, in the same sampling time, agent $A_2$ is included in the existing coalition. Afterwards, the optimal solutions are re-calculated, for both the coalition and the agents outside the coalition, and the feasibility is re-evaluated. Following this reasoning, in the extreme case, all the agents from the network can be gradually included in a single coalition. This represents the particular case of a centralized optimization problem. Inside a coalition $C$, the following optimization problem is minimized:

$$
\begin{align*}
    J_C(x_C^0) &= \min_{u_C^0, \ldots, u_C^{N_p-1}} \max_{x_C^0, \ldots, x_C^{N_p-1}} \\
    &\quad \|R_{w_C} x_C^{\text{max}}\|_1 + \sum_{i=0}^{N_p-1} \|x_C^i\|_1 + \|R_{u_C} u_C^i\|_1 + \|x_C^{N_p}\|_1 \\
    \text{s. t.} \quad &x_C^{\text{min}} \leq x_C^i \leq x_C^{\text{max}}, \quad l = 1, \ldots, N_p - 1 \\
    &x_C^{N_p} \in \Omega_C \\
    &u_C^{\text{min}} \leq u_C^i \leq u_C^{\text{max}}, \quad l = 0, \ldots, N_p - 1 \\
    &w_C^{\text{min}} \leq w_C^i \leq w_C^{\text{max}} \\
\end{align*}
$$

(7)

where, the weight matrices $R_{w_C}$ and $R_{u_C}$ are block diagonal; $\Omega_C$ is the aggregated terminal set; $x_C^0$ is the aggregated vector containing the corresponding initial state values; $x_C^{\text{max}}$, $u_C^{\text{max}}$, $w_C^{\text{max}}$ are the aggregated maximum limits for the state, input and the uncertainty sequences, respectively; $x_C^{\text{min}}$, $u_C^{\text{min}}$ are the aggregated minimum limits for the state and input sequences, respectively; $u_C^i = \{u_C^{0,i}, \ldots, u_C^{N_p-1,i}\}$,
\[ w_C^l = \{ w_C^0, \ldots, w_C^{N_p-1} \} \] and \( x_C^l \) are the aggregated input, uncertainty and state trajectories of the coalition \( C \); the optimization variable \( \hat{x}_{C_{\text{max}}} \) is the aggregated self-imposed limit for the aggregated state sequence.

- Between neighbouring agents, only the self-imposed optimization variable \( \hat{x}_{\text{max}} \) from (3) is shared. In this manner, although the sub-systems are state-coupled, the coupling signals are considered as local uncertainties, and a robust optimization problem is defined using the shared information as the uncertainty limit. With this value received from the neighbour, each local agent searches for a robust positively invariant set in the predefined table, computed in the initialization phase of the algorithm (see Section 2). If there is no pre-computed terminal region \( \Omega_i \) which can include the received uncertainty, then the local optimization problem becomes infeasible and the coalition procedure is started.

In what follows, a summarized pseudo-code for the coalitional DMPC, which uses the invariant sets as terminal regions is provided. Starting from the algorithm provided in [18], the following novel algorithm is obtained for the current system configuration, namely state-coupled sub-systems, which at each iteration must exchange the state uncertainty polytope.

**Remark 1.** In Algorithm 2, step 6, the local feasibility is verified for all agents in the network. When more than one agent has an infeasible local solution, a sub-unitary random coalition priority is given to each one of them. This means that, to avoid conflicts and subjectivity, all agents in need of a coalition, i.e., due to infeasibility, have an equal chance of initially entering in a coalition. However, after this random coalition initialization start-up, the methodology sequentially increases the size of the coalition when needed, always starting from coalitions of 2 agents.

**Algorithm 2**

Initialization: For each agent \( A_i \), \( \forall i \in \mathcal{N} \), compute a table \( T_i \), with potential terminal sets \( \Omega_i \).

At each sampling time \( k \), each agent \( A_i \), \( \forall i \in \mathcal{N} \), receives the local state value and performs the following steps:

1. Computes the uncertainty polytope using default limit values for the state-uncertainty constraints:
   \[ W_i = A_i x_{i-1}^{0} \]
   \[ X_{i-1} = [x_{i-1}^{0}, -x_{i-1}^{0}] \]

2. Searches in the predefined table \( T_i \) for a terminal set \( \Omega_{0,i} \) that includes the default uncertainty \( W_i \subseteq \Omega_{0,i} \).

3. Solves the local optimization problem (3) and obtains the optimal values \( U_{i,0}^*, x_{i,0}^{\text{max}} \) using the default values \( \Omega_i = \Omega_{0,i} \) for the terminal set and the uncertainty constraint limit (\( x_{i,0}^{\text{max}} = x_{i-1}^{\text{max}} \)).

4. Broadcasts to its successor the local optimization value \( x_{i,0}^{\text{max}} \) and receives the corresponding value \( x_{i-1,0}^{\text{max}} \) from its predecessor.

5. Repeats Steps 1–3 using the uncertainty constraint value received in Step 4.

6. Checks the feasibility of the local optimization problem:

   - If the optimization problem from Step 5 is feasible:
     then: Coalitions between agents are not necessary. Each local agent \( A_i \) sends to its sub-system \( S_i \), the first value from the optimal trajectory \( U_{i}^* \);
     else: Coalitions between agents are necessary. The coalition model is defined as in (5), by aggregating the models of the sub-systems involved in the coalition. Moreover, the optimization problem (3) is redefined in terms of the coalition and is solved.

   - If all the optimization problems (inside and outside the coalition) are feasible:
     then: The existing coalition was successful and can be dissolved after every sub-system \( S_i \) receives the first value from the optimal trajectory \( U_{i}^* \);
     else: The existing coalition was not successful. Another agent must be included in the existing coalition.

7. End algorithm.
Remark 2 ([18,19]). Within Algorithm 2, the feasibility of the coalition control problem is ensured because $\mathcal{W}_C \subseteq \prod_{i \in C} \mathcal{W}_i, \mathcal{U}_C = \prod_{i \in C} \mathcal{U}_i$, and $\mathcal{Q}_C = \prod_{i \in C} \mathcal{Q}_i$. Keeping the coalition stable is guaranteed by the terminal constraint set, calculated as the Minkowski sum of the polytopes for each individual agent of the coalition. As long as initially all of the local optimization problems can be solved in a decentralized fashion, the algorithm is recursively feasible.

5. Simulation Results

In this section, the C-DMPC methodology is tested in simulation for a vehicle platooning application described in Section 5.1 and the obtained results are provided in Section 5.2. Moreover, the string stability analysis of the platoon is provided in Section 5.3.

5.1. Platooning Application Setup

In this subsection, the C-DMPC methodology is tested in simulation for a vehicle platooning application described in Section 5.1 and the obtained results are provided in Section 5.2. Moreover, the string stability analysis of the platoon is provided in Section 5.3.

5.1. Platooning Application Setup

In this subsection, the C-DMPC methodology is tested in simulation for a $N = 4$ vehicle platooning application, as described in [39]. Each vehicle from the platoon is modelled with the following continuous-time dynamical model [40]:

$$
\begin{bmatrix}
\Delta d_i \\
\Delta v_i \\
a_i
\end{bmatrix}
= \begin{bmatrix}
0 & 1 & -h \\
0 & 0 & 1 \\
0 & 0 & -1/\tau_i
\end{bmatrix}
\begin{bmatrix}
\Delta d_i \\
\Delta v_i \\
a_i
\end{bmatrix}
+ \begin{bmatrix}
0 \\
0 \\
1/\tau_i
\end{bmatrix}
u_i + \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 1 \\
0 & 0 & 0
\end{bmatrix}x_{i-1}
$$

(8)

where $\Delta d_i(t) = d_i(t) - d_{i+1}(t)$ is the distance between platoon vehicles, and $d_{i+1}(t) = r - hv_i(t)$ is the desired relative distance, computed taking into account the standstill distance $r$ and the time headway value $h$. The relative velocity between two consecutive vehicles is $\Delta v_i(t) = v_{i-1}(t) - v_i(t)$, and the relative acceleration is $\Delta a_i(t) = a_{i-1}(t) - a_i(t)$. Note that the vehicle’s model (8) is coupled through the states. The time constant of the dynamics for the vehicle’s engine is $\tau_i$, while $u_i$ is its input, defined as the desired acceleration for the vehicle. In Figure 1, a schematic diagram for a vehicle platooning application is provided, in which, the leader denoted $V_0$ travels with a constant velocity and has $\dot{a}_0 = 0$ and $u_0 = 0$, and is considered a virtual leader which is not optimized. Thus, in this paper, we control only the follower vehicles from the platoon, denoted with $V_i, \forall i \in \{1, \ldots, 4\}$.

![Figure 1. Schematic diagram of a vehicle platoon.](image)

For simulation purposes, all followers are considered identical, with $\tau_i = 0.1, \forall i \in \{1, \ldots, 4\}$, and the headway time value $h = 1.5$. The vehicle’s model (8) was discretized with the sampling period $T_s = 0.1$ seconds.

The optimization parameters used in the cost function (3) are: the prediction horizon $N_p = 26$ samples; the input and the uncertainty weights $R_{u_i} = 0.1$ and $R_{w_i} = 1$, respectively, $\forall i \in \{1, \ldots, 4\}$. These values were empirically selected to achieve the best results in the C-DMPC algorithm. This means that, the feasibility of the algorithm must be ensured at each sampling time, for one of the possible communication topologies.

The numerical values for the limit constraints described in (2) are as follows:

$$
\begin{align*}
\{ u_i^{\min}, u_i^{\max} \} & = \{ -5, 5 \} & \{ y_i^{\min}, y_i^{\max} \} & = \{ -7, 7 \} \\
\{ w_i^{\min}, w_i^{\max} \} & = \{ -1, 1 \} & \forall i \in \{1, \ldots, 4\}
\end{align*}
$$
Please note that, in order to compute the table with invariant sets, these values were taken into account.

To compute the feedback control laws \( K_i, \forall i \in \{1, \ldots, 4\} \), using the vehicle model (8), two case-studies were analysed, corresponding to the two methods that were used, namely:

- **Case 1**: classical state-feedback control, based on the Ackermann’s formula [36], obtaining:

\[
K = K_1 = K_2 = K_3 = K_4 = [0.5374 \quad 0.0263 \quad 0.3628].
\] (9)

Please note that, this feedback gain matrix was obtained imposing identical desired closed-loop transient performance for each sub-system, namely an overshoot value of 5% and settling time of five time units.

- **Case 2**: minimization of a linear-quadratic cost function, by solving a discrete-time Riccatti Equation [37]:

\[
\tilde{K} = \tilde{K}_1 = \tilde{K}_2 = \tilde{K}_3 = \tilde{K}_4 = [0.4044 \quad 0.6045 \quad -0.1562].
\] (10)

It is worthy to mention that, these matrices were computed using the same weight matrix as in the solved optimization problem, i.e., \( K_{ui} = 5 \).

The invariant sets obtained for sub-system \( S_1 \) using Algorithm 1, from Section 3 are depicted in Figures 2 and 3. Those sets were computed using the following numerical values:

- \( u_{\text{max}} = 10 \), \( w_{\text{max}} = 10 \), \( w_{\text{min}} = 0.1 \), \( \text{step}_a = \text{step}_b = 0.5 \). The polytopes computed using the feedback matrix \( K \) from Case 1 are denoted with \( \Omega_i \), and plotted in Figure 2. Similarly, the polytopes calculated using the feedback matrix \( \tilde{K} \) from Case 2 are denoted with \( \tilde{\Omega}_i \) and depicted in Figure 3. In both figures, the largest invariant set (i.e., corresponding to the maximum input constraint value \( a = u_{\text{max}} = 10 \) and the minimum uncertainty \( \beta = 0.1 \)) is depicted in red colour. As expected, the larger possible invariant set that can be used in the terminal constraint definition includes the other smaller invariant sets. This is especially visible in Figure 3, in which the smaller polytopes, obtained when the input constraint is decreased are gradually “cropped” from the dimension of the red polytope. Both Figures 2 and 3 provide a graphical representation for the invariant set table computed in the initialization phase of the algorithm, i.e., for each cell in the table, a coloured polytope is depicted. According to the design of Algorithm 1 from Section 3, several invariant sets are computed, for different parametrizations of the input and uncertainty constraints limits, via \( a \) and \( \beta \) values. Thus, the larger invariant set polytope (i.e., corresponding to the maximum input constraint value \( a = u_{\text{max}} = 10 \) and the minimum uncertainty \( \beta = 0.1 \)) is denoted with \( \Omega_1 \) (see Figures 2 and 3). However, as the input limit decreases, also the invariant sets have smaller dimensions, which are included in the larger set. Note that, the computed invariant sets are three dimensional, because the state variable for the vehicle model (8) has three values, and can be plotted as convex hull (ref. Figures 2 and 3).

**Remark 3.** Algorithm 1, from Section 3 was implemented in MATLAB R2020b using the YALMIP toolbox [41]. For each input, state and uncertainty constraint set values, a maximum of 100 iterations were computed to search for the robust positively invariant set.

**Remark 4.** The computational complexity of Algorithm 1, is related to the number of iterations required to compute each robust positively invariant set \( \Omega_i \) from table \( T_i \). This value gives the invariance order for each \( \Omega_i \). Although this is computed only once and outside of the procedure which solves the optimization (see Initialization from Algorithm 2), the necessary computational time is non-trivial. This is influenced by the used parametrization, which should be chosen according to the sub-system’s constraint limits (i.e., the values \( a \) and \( \beta \), and their corresponding step size values \( \text{step}_a = \text{step}_b \)). Furthermore, the feedback gain matrix \( K \) has a significant impact on the algorithm complexity, which can be seen in the invariance orders for each \( \Omega_i \) set. Hence, under the same parametrization, using the feedback matrix \( K \) from Case 1, 16 invariant sets were obtained, with invariance orders ranging from 14 to 45 iterations. Furthermore, using the feedback matrix \( \tilde{K} \)
from Case 2, 17 invariant sets were obtained, with invariance orders ranging from 3 to 36 iterations. Note that, for simplicity, only the first 7 polytopes, in the descending order with respect to the $\alpha$ value, denoted with $\Omega_i$, $\forall i \in \{1, \ldots, 7\}$, are plotted.

Figure 2. Case 1—Depiction of the predefined invariant sets corresponding to sub-system $S_1$, computed for uncertainty constraint limit value $\beta = 0.1$.

Figure 3. Case 2—Depiction of the predefined invariant sets corresponding to sub-system $S_1$, computed for uncertainty constraint limit value $\beta = 0.1$. 
5.2. Illustrative Results

In the local optimization problem formulated in (3), the control goal was to regulate all local states to zero. From a vehicle platooning perspective, in which each vehicle is modelled with (8), this results in a vehicle platoon which travels with a constant speed and implicitly with 0 acceleration, and maintains a constant desired inter-vehicle distance. A multitude of tests were performed on the homogeneous platoon defined in Section 5.1. These were conducted using both homogeneous and heterogeneous initial conditions $x^0$, for each vehicle $\forall i \in \{1, \ldots, 4\}$, with numerical values selected within the imposed constraint limits. However, for brevity, the most representative tests are provided, with the following initial values:

**Remark 5.** Two simulations were performed for each test, using the invariant set tables $T_i$, computed with the feedback matrix $K$ from (9) and $\tilde{K}$ from (10), respectively. The simulation results obtained using the invariant sets $\Omega_i$, computed with the feedback matrix $\tilde{K}$ from Case 2, were almost identical to the ones computed with the feedback matrix $K$ from Case 1, and for brevity were not included in the paper.

The performance of the proposed C-DMPC algorithm was studied for all the simulations mentioned in Table 1, and is given in Table 2. This evaluation was performed computing the cumulative cost index in norm-1, using the state vectors of each vehicle $V_i$, $\forall i \in \{1, \ldots, 4\}$, from the platoon, i.e., $J_{V_i} = \|\Delta d_i\|_1 + \|\Delta v_i\|_1 + \|a_i\|_1$. Moreover, for an overall view, the cumulative cost index for each simulated test $Test_j$, $\forall j \in \{1, \ldots, 8\}$, is also computed, i.e., $J_{Test_j} = \sum_{i=1}^{4} J_{V_i}$. The computational time for each simulation test is also provided. The consistence of the performance indices values effectively illustrate the robustness of the proposed C-DMPC algorithm.

| Test  | $x_1^0$ | $x_2^0$ | $x_3^0$ | $x_4^0$ |
|-------|---------|---------|---------|---------|
| Test1 | [1.5 0 0]^T | [1.5 0 0]^T | [1.5 0 0]^T | [1.5 0 0]^T |
| Test2 | [2 0 0]^T | [2 0 0]^T | [2 0 0]^T | [2 0 0]^T |
| Test3 | [2.5 0 0]^T | [2.5 0 0]^T | [2.5 0 0]^T | [2.5 0 0]^T |
| Test4 | [3.5 0 0]^T | [3.5 0 0]^T | [3.5 0 0]^T | [3.5 0 0]^T |
| Test5 | [2 0 0]^T | [1.5 0 0]^T | [1 0 0]^T | [0.5 0 0]^T |
| Test6 | [2 0 0]^T | [2 0 0]^T | [1 0 0]^T | [1 0 0]^T |
| Test7 | [3 0 0]^T | [2.5 0 0]^T | [2 0 0]^T | [1.5 0 0]^T |
| Test8 | [3.5 0 0]^T | [3 0 0]^T | [2.5 0 0]^T | [2 0 0]^T |

| Test | $J_{V_1}$ | $J_{V_2}$ | $J_{V_3}$ | $J_{V_4}$ | $J_{Test_j}$ | Sim Time (s) |
|------|----------|----------|----------|----------|--------------|--------------|
| Test1 | 0.0357 | 0.0371 | 0.0354 | 0.0367 | 0.1449 | 2.7925 |
| Test2 | 0.0476 | 0.0496 | 0.0484 | 0.0493 | 0.1950 | 2.7627 |
| Test3 | 0.0620 | 0.0635 | 0.0635 | 0.0671 | 0.2561 | 3.1780 |
| Test4 | 0.0869 | 0.0900 | 0.0914 | 0.0948 | 0.3631 | 3.2602 |
| Test5 | 0.0476 | 0.0375 | 0.0241 | 0.0132 | 0.1224 | 3.3953 |
| Test6 | 0.0476 | 0.0496 | 0.0249 | 0.0245 | 0.1467 | 2.8173 |
| Test7 | 0.0745 | 0.0623 | 0.0528 | 0.0444 | 0.2340 | 3.0752 |
| Test8 | 0.0834 | 0.0747 | 0.0620 | 0.0340 | 0.2540 | 3.1732 |

**Remark 6.** The proposed algorithm was tested in simulation and was designed to ensure satisfactory control performances for a vehicle platooning application. The simulations were performed using MATLAB R2020b on Windows 10, 64-bit Operating System with a laptop Intel Core i7-9850H CPU @ 2.60 GHz, 2460 and 16 GB RAM. Thus, the algorithm was not yet optimized to be executed on embedded devices, and to be tested in a real-time setup, but this is a subject of future work.
Hereafter, for brevity, only the simulation results for two tests from Table 1 are presented, namely starting from: (i) the homogeneous initial values from Test4, and (ii) the heterogeneous initial values from Test8.

The relative distance error, the relative speed and the acceleration for each follower vehicle, respectively, obtained for the initial conditions from Test4 (ref. Table 1), are depicted in Figures 4–6. Furthermore, the same variables are depicted in Figures 7–9, obtained for the initial conditions from Test8 (ref. Table 1).

As expected, at the end of the simulation scenario, all the states and implicitly the inputs (see Figures 10 and 11, respectively) reach the desired zero value.

![Figure 4. Test4—Distance error for each platoon vehicle.](image)

![Figure 5. Test4—Relative speed for each platoon vehicle.](image)
Figure 6. Test 4—Acceleration for each platoon vehicle.

Figure 7. Test 8—Distance error for each platoon vehicle.

Figure 8. Test 8—Relative speed for each platoon vehicle.
From the C-DMPC point of view, a platoon consisting of 4 vehicles uni-directionally coupled in a leader-follower communication architecture, has eight possible communication frameworks, including coalitions of 2, 3 or 4 agents defined in Table 3 [18].
As previously mentioned, in the proposed C-DMPC algorithm, all agents first solve a robust min-max non-cooperative DMPC problem with state uncertainty, starting from given initial conditions. This can be seen as $C_{\emptyset}$ active, in which all agents are outside a coalition. If one or more local solutions are infeasible, then the coalition procedure starts and coalitions between different agents are formed.

When analysing both Figures 12 and 13, one can remark that only in the beginning of the simulation test, i.e., for the first 12 samples (see Figure 12) and, respectively, 11 samples (see Figure 13), the coalitions are activated.
This is due to the fact that for the particular initial condition from the simulation test chosen for the platoon, some of the local problems are infeasible. Our proposed C-DMPC algorithm is designed such that, at each sampling time, when a local problem is infeasible, i.e., the min-max DMPC algorithm fails to reach a solution, a coalition is activated. Moreover, the algorithm always starts with the smallest coalition size, i.e., including two agents, and when needed, increases its dimension by sequentially including other neighbours, until the largest possible size is reached, i.e., see the activation of $C_{1234}$ at time step $k = 1$ in Figure 12. After the feasible solution is reached, the current coalition deactivates, and the procedure is repeated at the next sampling time. This is the reason that in Figure 12, at time step $k = 2$, a smaller size coalition $C_{23}$ between agents $A_2$ and $A_3$ is activated. Coincidentally, in Figure 12, coalition $C_{23}$ is consecutively activated at time steps $k = 2, 3$ and $4$, whereas coalition $C_{34}$ between $A_3$ and $A_4$ is activated from time samples $k = 5$ to $k = 12$. Afterwards, since both the acceleration state and the control input almost reach the desired zero value (see the detailed plot from Figures 6 and 10), there is no need for the coalition activation. In this case, the uncertainty received from the predecessor vehicle is included in one of the predefined terminal sets, and the local problem is feasible. Hence, starting from time instant $k = 13$ until the end of the simulation, the control architecture for all the platoon vehicles is min-max non-cooperative DMPC (or $C_∅$). For a better visualisation, only the first 15 time samples are plotted in Figure 12. An agent inside a coalition is depicted with a red star marker within a blue circle, whereas for one outside a coalition, the corresponding blue circle is empty. When analysing Figure 13, the same reasoning is used and one can remark that coalition $C_{23}$ is activated from time samples $k = 1$ to $k = 3$, while coalition $C_{34}$ is activated from time samples $k = 4$ to $k = 11$. After that time-instant, no more coalitions are activated.

5.3. String Stability Analysis

The string stability of the vehicle platoon, which ensures that the inter-vehicle distance error does not amplify towards the end of the platoon is analysed using the formulation given in [34]:

$$\|\Delta d_i\|_\infty \leq \epsilon_i \|\Delta d_{i-1}\|_\infty$$

(11)

where $\Delta d_i$ and $\Delta d_{i-1}$ are the distance errors for two consecutive vehicles from the platoon, and $\epsilon_i \in (0, 1)$ is a parameter.

To verify that the platoon is string stable, the property (11) was used a posteriori, similarly with the work in [35], by analysing the corresponding $\Delta d_i$ values obtained for each vehicle from the platoon. This was performed by computing the $\epsilon$ values for every two consecutive vehicles in the platoon, which according to (11), must be smaller or equal to 1, to ensure that the errors do not propagate towards the end of the platoon. Since the simulated platoon does not have a leader vehicle, the first investigation was performed between vehicles $V_2$ and $V_3$ by firstly computing $\epsilon_3$. Next, $\epsilon_4$ was computed between vehicles $V_3$ and $V_4$.

All the simulation tests starting from the initial conditions provided in Table 1 were investigated, and the results are given in Table 4.

| Test   | $\epsilon_3$ | $\epsilon_4$ |
|--------|--------------|--------------|
| Test1  | 1.0000       | 1.0000       |
| Test2  | 1.0000       | 1.0000       |
| Test3  | 1.0000       | 1.0000       |
| Test4  | 1.0000       | 1.0000       |
| Test5  | 0.6667       | 0.5000       |
| Test6  | 0.5000       | 1.0000       |
| Test7  | 0.8000       | 0.7500       |
| Test8  | 0.8333       | 0.4000       |
However, one must take into account that our results are obtained in the C-DMPC framework. This means that when a coalition is activated, the vehicles inside the coalition must be viewed as a single entity, since the models are aggregated in the coalitional optimization problem. Thus, according to the coalitions activated in Test 8 simulation, when $C_{23}$ is active, the distance error for $A_3$ is not taken into account, thus analysing only the states for vehicles 1, 2 and 4. Moreover, when $C_{34}$ activates, only the states for vehicles 1, 2 and 3 are meaningful, since $A_4$ is inside the coalition, and ‘coupled’ with $A_3$.

Hence, the string stability property for the proposed C-DMPC suitable for a platooning application is satisfied in all the investigated tests.

Remark 7. When analysing the string stability of a platoon controlled with a coalitional methodology, one must take into account the coalitions which are formed between the vehicles, which implicitly guarantee the feasibility of all the optimization problems.

6. Conclusions
In this paper, a coalitional distributed model predictive (C-DMPC) methodology, suitable for a vehicle platooning application was proposed. Details regarding the computation of the maximal robust positively invariant sets, which were used as terminal regions in the C-DMPC algorithm were also provided. The control strategy was tested for a vehicle platooning application, consisting of a virtual leader and four follower vehicles, with a leader-follower uni-directional communication topology. The simulation results show the efficiency of the proposed coalitional DMPC algorithm, in which the coalitions between different agents are formed when needed (i.e., the local feasibility of the solution is lost). Otherwise, the control topology between vehicles is robust min-max non-cooperative DMPC with state uncertainties.

Future work will test the proposed C-DMPC algorithm in an experimental setup using a platoon of mobile robots.

7. Materials and Methods
The simulations from this work were performed using MATLAB R2020b on Windows 10, 64-bit Operating System on a laptop (Intel Core i7-9850H CPU @ 2.60 GHz and 16 GB RAM). The optimizations were implemented using the YALMIP toolbox [41].

Author Contributions: Conceptualization, A.M. and C.-F.C.; methodology, A.M.; software, A.M.; validation, A.M. and C.-F.C.; formal analysis, A.M.; investigation, A.M.; resources, A.M.; data curation, A.M.; writing—original draft preparation, A.M.; writing—review and editing, A.M. and C.-F.C.; visualization, A.M.; supervision, C.-F.C.; project administration, A.M. and C.-F.C.; funding acquisition, A.M and C.-F.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by a publications grant of the TUIASI, project number GI/P7/2021.

Institutional Review Board Statement: Not applicable

Informed Consent Statement: Not applicable

Data Availability Statement: Not applicable

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Abbreviations
The following abbreviations are used in this manuscript:

MPC Model Predictive Control
DMPC Distributed Model Predictive Control
C-DMPC Coalitional Distributed Model Predictive Control
References

1. Yu, S.; Hirche, M.; Huang, Y.; Chen, H.; Allgöwer, F. Model predictive control for autonomous ground vehicles: A review. *Auton. Intell. Syst.* 2021, 1, 4. [CrossRef]

2. Jin, L.; Cicic, M.; Johansson, K.H.; Amin, S. Analysis and Design of Vehicle Platooning Operations on Mixed-Traffic Highways. *IEEE Trans. Autom. Control* 2021, 66, 4715–4730. [CrossRef]

3. Caruntu, C.F.; Braescu, C.; Maxim, A.; Rafaila, R.C.; Tiganasu, A. Distributed model predictive control for vehicle platooning: A brief survey. In Proceedings of the 20th International Conference on System Theory, Control and Computing, Sinaia, Romania, 13–15 October 2016; pp. 644–650.

4. Wang, Z.; Wu, G.; Barth, M.J. A Review on Cooperative Adaptive Cruise Control (CACC) Systems: Architectures, Controls, and Applications. In Proceedings of the 21st International Conference on Intelligent Transportation Systems, Maui, HI, USA, 4–7 November 2018; pp. 2884–2891.

5. Fakhfakh, F.; Tounsi, M.; Mosbah, M. Vehicle Platooning Systems: Review, Classification and Validation Strategies. *Int. J. Netw. Distrib. Comput.* 2020, 8, 203–213. [CrossRef]

6. Wang, J.; Zheng, H.; Guo, J.; Fan, J.; Li, K. Distributed adaptive robust H-infinity control of intelligent-connected electric vehicles platooning subject to communication delay. *IET Intel. Transp. Syst.* 2021, 15, 699–711. [CrossRef]

7. Xiong, X.; Sha, J.; Jin, L. Optimizing coordinated vehicle platooning: An analytical approach based on stochastic dynamic programming. *Transp. Res. Part B—Methodol.* 2021, 150, 482–502. [CrossRef]

8. Sun, X.; Yin, Y. Decentralized game-theoretical approaches for behaviorally-stable and efficient vehicle platooning. *Transp. Res. Part B—Methodol.* 2021, 153, 45–69. [CrossRef]

9. Dolk, V.S.; Ploeg, J.; Heemels, W.P.M.H. Event-Triggered Control for String-Stable Vehicle Platooning. *IEEE Trans. Intell. Transp. Syst.* 2017, 8, 3846 – 3850. [CrossRef]

10. Talpi, T.; Kiran, U. A Modified Distributed Model Predictive Control Approach for Handling Vehicular Platoon Communication Problems. In Proceedings of the IEEE 23rd International Conference on Intelligent Transportation Systems, Rhodes, Greece, 20–23 September 2020; pp. 1–6.

11. Lan, J.; Zhao, D. Min-Max Model Predictive Vehicle Platooning with Communication Delay. *IEEE Trans. Veh. Technol.* 2020, 69, 12570–12584. [CrossRef]

12. Liu, P.; Ozguner, U.; Zhang, Y. Distributed MPC for cooperative highway driving and energy-economy validation via microscopic simulations. *Transp. Res. Part C Emerg. Technol.* 2017, 77, 80–95. [CrossRef]

13. Wang, P.; Deng, H.; Zhang, J.; Wang, L.; Zhang, M.; Li, Y. Model Predictive Control for Connected Vehicle Platoon Under Switching Communication Topology. *IEEE Trans. Intell. Transp. Syst.* 2021, 1–14. [CrossRef]

14. Scattolini, R. Architectures for distributed and hierarchical Model Predictive Control—A review. *J. Process Control* 2009, 19, 723–731. [CrossRef]

15. Christofides, P.D.; Scattolini, R.; Muñoz de la Peña, D.; Liu, J. Distributed model predictive control: A tutorial review and future research directions. *Comput. Chem. Eng.* 2013, 51, 21–41. [CrossRef]

16. Fele, F.; Maestre, J.M.; Camacho, E.F. Coalitional control: Cooperative game theory and control. *IEEE Control Syst.* 2017, 37, 53–69. [CrossRef]

17. Chanfreut, P.; Maestre, J.M.; Muros, F.J.; Camacho, E.F. A Coalitional Control Scheme with Topology-Switchings Convexity Guarantees. In Proceedings of the 58th Conference on Decision and Control, Nice, France, 11–13 December 2019; pp. 1096–1101.

18. Maxim, A.; Caruntu, C.F. A Coalitional Distributed Model Predictive Control Perspective for a Cyber-Physical Multi-Agent Application. *Sensors* 2021, 21, 4041. [CrossRef] [PubMed]

19. Maxim, A.; Maestre, J.M.; Caruntu, C.F.; Lazar, C. Robust coalitional distributed model predictive control algorithm with stability via terminal constraint. In Proceedings of the IEEE Conference on Control Technology and Applications, Copenhagen, Denmark, 21–24 August 2018; pp. 964–969.

20. Camacho, E.F.; Bordons, C. *Model Predictive Control*; Springer: Berlin, Germany, 1999.

21. Maciejowski, J.M. *Predictive Control with Constraints*; Prentice-Hall, Pearson Education Limited: Harlow, UK, 2002.

22. Cannon, M.; Deshmukh, V.; Kouvaritakis, B. Nonlinear model predictive control with polytopic invariant sets. *Automatica* 2003, 39, 1487–1494. [CrossRef]

23. Mayne, D.Q. Control of Constrained Dynamic Systems. *Eur. J. Control* 2001, 7, 87–99. [CrossRef]

24. Limon, D.; Alamo, T.; Salas, F.; Camacho, E.F. Input to state stability of min–max MPC controllers for nonlinear systems with bounded uncertainties. *Automatica* 2006, 42, 797–803. [CrossRef]

25. Mayne, D.Q.; Seron, M.M.; Raković, S.V. Robust model predictive control of constraint linear systems with bounded disturbances. *Automatica* 2005, 41, 219–224. [CrossRef]

26. Decardi-Nelson, B.; Liu, J. Computing robust control invariant sets of constrained nonlinear systems: A graph algorithm approach. *Comput. Chem. Eng.* 2021, 150, 107177. [CrossRef]

27. Gupta, A.; Köroğlu, H.; Falcone, P. Computation of robust control invariant sets with predefined complexity for uncertain systems. *Int. J. Robust Nonlinear Control* 2021, 31, 1674–1688. [CrossRef]

28. Xue, B.; Wang, Q.; Zhan, N.; Fränzle, M. Robust invariant sets generation for state-constrained perturbed polynomial systems. In Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control, Montreal, QC, Canada, 16–18 April 2019; pp. 128–137.
29. Rungger, M.; Tabuada, P. Computing Robust Controlled Invariant Sets of Linear Systems. *IEEE Trans. Autom. Control* 2017, 62, 3665–3670. [CrossRef]
30. Esterhuizen, W.; Aschenbruck, T.; Streif, S. On maximal robust positively invariant sets in constrained nonlinear systems. *Automatica* 2020, 119, 109044. [CrossRef]
31. Aschenbruck, T.; Esterhuizen, W.; Streif, S. Transient stability analysis of power grids with admissible and maximal robust positively invariant sets. *at—Automatisierungstechnik* 2020, 68, 1011–1021. [CrossRef]
32. Schulze Darup, M.; Schaich, R.M.; Canon, M. How scaling of the disturbance set affects robust positively invariant sets for linear systems. *Int. J. Robust Nonlinear Control* 2016, 27, 3236–3258. [CrossRef]
33. Ren, Y.; Er, M.J.; Sun, G. Switched systems with average dwell time: Computation of the robust positive invariant set. *Automatica* 2017, 85, 306–313. [CrossRef]
34. Dunbar, W.B.; Caveney, D.S. Distributed receding horizon control of vehicle platoons: Stability and string stability. *IEEE Trans. Autom. Control* 2012, 57, 620–633. [CrossRef]
35. Pauca, O.; Maxim, A.; Caruntu, C.F. Control Architecture for Cooperative Autonomous Vehicles Driving in Platoons at Highway Speeds. *IEEE Access* 2021, 9, 153472 – 153490. [CrossRef]
36. Ogata, K. *Modern Engineering Control*, 5th ed.; Prentice Hall: Upper Saddle River, NJ, USA, 2010.
37. Kwakernaak, H.; Sivan, R. *Linear Optimal Control Systems*; John Wiley & Sons, Inc.: New York, NY, USA, 1972.
38. Maestre, J.M.; Muñoz de la Peña, D.; Camacho, E.F.; Alamo, T. Distributed model predictive control based on agent negotiation. *J. Process Control* 2011, 21, 685–697. [CrossRef]
39. Chanfreut, P.; Keijzer, T.; Ferrari, R.M.; Maestre, J.M. A Topology-Switching Coalitional Control and Observation Scheme with Stability Guarantees. *IFAC PapersOnLine* 2020, 53, 6477–6482. [CrossRef]
40. Zhu, Y.; He, H.; Zhao, D. LMI-Based Synthesis of String-Stable Controller for Cooperative Adaptive Cruise Control. *IEEE Trans. Intell. Transp. Syst.* 2021, 21, 4516–4525. [CrossRef]
41. Löfberg, J. YALMIP: A Toolbox for Modeling and Optimization in MATLAB. In Proceedings of the IEEE International Symposium on Computer Aided Control Systems Design, New Orleans, LA, USA, 2–4 September 2004; pp. 284–289.