MASTER

Decentralized and distributed model predictive control of vehicle platoons

Shi, S.

Award date:
2017

Link to publication
Decentralized and Distributed Model Predictive Control of Vehicle Platoons
Declaration concerning the TU/e Code of Scientific Conduct for the Master’s thesis

I have read the TU/e Code of Scientific Conduct¹.

I hereby declare that my Master’s thesis has been carried out in accordance with the rules of the TU/e Code of Scientific Conduct

Date 15-02-2017

Name Shenglong Shi

ID-number 0925030

Signature

Shenglong Shi

Submit the signed declaration to the student administration of your department.

¹ See: http://www.tue.nl/en/university/about-the-university/integrity/scientific-integrity/
The Netherlands Code of Conduct for Academic Practice of the VSNU can be found here also.
More information about scientific integrity is published on the websites of TU/e and VSNU

January 15 2016
Abstract

This thesis considers decentralized control and distributed control for vehicle platoons and more generally networked systems in a chain structure by using model predictive control (MPC) algorithms. Additionally, automatic controller synthesis is also discussed when the topology of the vehicle platoon changes.

The distributed models of the vehicle platoon are coupled through the input of the preceding vehicles. In the decentralized scheme, no communication among vehicles is available and thus the coupled input is regarded as unknown disturbance. Then, two robust MPC (RMPC) algorithms, i.e. [1] [2], are used to solve the decentralized control problem, which leads to two different decentralized model predictive control (DeMPC) schemes. It is demonstrated by simulation that the decentralized control problem can be solved by both DeMPC algorithms.

In the distributed control problem, communication among vehicles becomes available and thus the two DeMPC schemes are modified to incorporate the communication, which leads to two distributed model predictive control (DMPC) schemes. In addition, the proof of recursive feasibility for the DMPC algorithms is provided. Simulation demonstrates that the distributed control problem can be solved by both DMPC algorithms.

In addition, the decentralized control is compared with the distributed control of vehicle platoons. Overall, it is shown that the each DMPC algorithm have a larger feasible region than its corresponding DeMPC. The cost is that the communication is required and the total computation time is increased.

For the automatic controller synthesis when the topology of a platoon changes, algorithms of controller synthesis are provided and demonstrated by simulation for the scenario where one vehicle joins a platoon.
Acknowledgements

I would like to express my gratitude to my supervisor Dr. Mircea Lazar for his patient supervision and helpful comments. I learned a lot of new knowledge from Dr. Lazar and more importantly, his scientific methods of working and his rigorous attitude toward work were the most important things I learned during my master program. I believe this experience will not only guide my career but also my personal life in the future.

I would also like to thank my colleagues from Control System group: Dr. Veaceslav Spinu, Dr. Alejandro Marquez Ruiz and Bahadir Saltik. They provided me a lot of helpful suggestions when I encountered questions during my graduation project.

Finally, my special thanks go to my family. I would not have made it here without them. My father always showed interest in my work and it was very nice to discuss with him about how my work went and what my work was. I would also like to thank my mom for everything and she was always there whenever I needed support.
# Contents

Abstract ii

Acknowledgements iii

List of Figures v

Abbreviations vi

## 1 Introduction

1.1 Background ................................................................. 1
   1.1.1 Control of large-scale systems and vehicle platoons .......... 1
   1.1.2 MPC for vehicle platoons ......................................... 3
   1.2 Outline and Contribution ........................................... 4

## 2 Preliminaries and Problem formulation

2.1 Preliminaries ............................................................ 6
   2.1.1 Mathematical preliminaries ....................................... 6
   2.1.2 System and control theory ....................................... 7
   2.1.3 Basics of robust MPC ............................................. 10
   2.1.4 Working mechanism of platoons ................................ 14
2.2 Problem formulation ................................................... 17
   2.2.1 Decentralized control problem ................................ 18
   2.2.2 Distributed control problem .................................... 18
   2.2.3 Distributed synthesis of plug and play operations .......... 19

## 3 Open-loop DeMPC and DMPC

3.1 Input scaling ................................................................ 21
3.2 Decentralized control with RMPC ................................... 22
   3.2.1 Stability and Recursive feasibility ............................ 23
3.3 Distributed control with DMPC ....................................... 25
3.4 Distributed synthesis for plug and play .......................... 28
   3.4.1 Parameter generation for distributed synthesis .............. 28
   3.4.2 Synthesis of a local controller ................................. 30
3.5 Numerical analysis ....................................................... 31
   3.5.1 Parameters for simulation ....................................... 32
   3.5.2 Constant distance .................................................. 32
   3.5.3 Velocity-dependent distance .................................... 34
3.5.4 Distributed synthesis ........................................... 37
3.6 Conclusion ......................................................... 38

4 Feedback DeMPC and DMPC ........................................... 40
  4.1 Decentralized control with feedback RMPC ......................... 40
    4.1.1 Input parameterization ....................................... 40
    4.1.2 MPC formulation ............................................. 41
  4.2 Distributed control with feedback DMPC .......................... 42
    4.2.1 Recursive feasibility under time-varying disturbance sets ...... 42
    4.2.2 Distributed MPC with updating of disturbance sets ........... 45
  4.3 Distributed synthesis for Plug and Play .......................... 47
  4.4 Numerical analysis ............................................... 47
    4.4.1 Parameters and implementation ................................ 48
    4.4.2 Simulation results ........................................... 48
  4.5 Comparison between feedback and open-loop MPC ................. 50
    4.5.1 Under the same prediction horizon .......................... 51
    4.5.2 Under different prediction horizons .......................... 51
    4.5.3 Computation types of two DMPC algorithms ................... 52
  4.6 Conclusion ......................................................... 52

5 Application issues .................................................... 54
  5.1 Platoon size ...................................................... 54
  5.2 Joining and Leaving ............................................. 55
  5.3 String stability .................................................. 55
  5.4 Choice of the algorithm ......................................... 57

6 Conclusions and recommendations ...................................... 59
  6.1 Conclusions ....................................................... 59
  6.2 Recommendations ................................................. 60

A Computation of control invariant sets ................................ 62
  A.1 Problem formulation ............................................. 62
  A.2 Simplified case .................................................. 64
    A.2.1 Invertible Q and A ......................................... 64

Bibliography ............................................................. 66
List of Figures

1.1 Flowchart of the thesis .............................................. 5
2.1 Two adjacent vehicles. .............................................. 15
3.1 The evolution of $e_p^2$ in 2-vehicle platoon with $h = 0$ and $C = \text{diag}(0.9, 0.9)$. 33
3.2 $X^2_h \sim \mathbb{D}^2$ when $h = 0$ under different $C$. .......................... 34
3.3 Simulation results for 5 vehicles when $h = 1$. .......................... 35
3.4 Feasible region for the 5th vehicle. .................................. 35
3.5 $e_p^5$ and $u^5$ with $h = 1$ and perturbed communication. ............... 37
3.6 Simulation of a vehicle joining a platoon when time=50 .................. 38
4.1 Simulation results for 3 vehicles with feedback MPC .......................... 49
5.1 Input profiles using open-loop DMPC when $h = 1.4$ .......................... 56
### Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPC</td>
<td>Model Predictive Control</td>
</tr>
<tr>
<td>RMPC</td>
<td>Robust Model Predictive Control</td>
</tr>
<tr>
<td>DeMPC</td>
<td>Decentralized Model Predictive Control</td>
</tr>
<tr>
<td>DMPC</td>
<td>Distributed Model Predictive Control</td>
</tr>
<tr>
<td>PI</td>
<td>Positively Invariant</td>
</tr>
<tr>
<td>RPI</td>
<td>Robust Positively Invariant</td>
</tr>
<tr>
<td>CI</td>
<td>Control Invariant</td>
</tr>
<tr>
<td>RCI</td>
<td>Robust Control Invariant</td>
</tr>
<tr>
<td>ISS</td>
<td>Input-to-state Stability</td>
</tr>
<tr>
<td>QP</td>
<td>Quadratic Program</td>
</tr>
<tr>
<td>ACC</td>
<td>Adaptive Cruise Control</td>
</tr>
<tr>
<td>CACC</td>
<td>Cooperative Adaptive Cruise Control</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

This work considers the decentralized model predictive control (DeMPC) and the distributed model predictive control (DMPC) of a vehicle platoon in longitudinal direction. The dynamics of the vehicle platoon can be regarded as a networked system of linear systems coupled through inputs. The control goal is to control the vehicles to achieve automatic following with safe distance. The platoon control problem has been extensively investigated in frequency domain [3] [4]. However, it is difficult to include constraints, e.g. collision avoidance constraints and input constraints, in the frequency domain design. Hence, we are interested in MPC algorithms for vehicle platooning which can consider the constraints explicitly. The scenario where no information can be communicated among vehicles motives us to use DeMPC scheme to solve the platoon control problem. If the communication among vehicles is available, DMPC scheme will be exploited to solve the platoon control problem.

1.1 Background

1.1.1 Control of large-scale systems and vehicle platoons

Due to the increasing ability of communicating, computing and sensoring, systems tend to interact with each other physically and virtually, which forms complex networked systems. The traditional centralized control architecture collects the information from all
Introduction

subsystems and tries to solve a large-scale control problem, e.g. a large-scale optimization problem in the case of MPC. This architecture requires intensive communication and computational resources, which is not practically feasible in some applications.

At the other extreme, every subsystem in the large-scale systems can have a local controller for itself and the controller only collects strictly local information from the subsystem, which is the so-called decentralized control architecture. An example of DeMPC can be found in [5]. In this case, communication is not required at all during the closed-loop operation and the overall computational effort is small. However, the overall performance can be much poorer than the centralized controller which has the full information of the whole large-scale system.

Distributed control is a compromise between the centralized control and the decentralized control. In this control architecture, every subsystem also has a local controller which, however, communicates with other controllers to improve the performance for the overall system compared to the decentralized architecture. Examples can be found in [6] [7]. Specifically, DMPC controllers for networked systems aim to reduce computational effort by replacing the central optimization problem from the central MPC with several small-scale optimization problems while obtaining comparable performance with the centralized design.

The vehicle platoon is an example of a large-scale system. It consists of a group of vehicles in a chain structure where the leader tracks a reference and the followers shall achieve automatic following while keeping a safe distance using the obtained controllers. In addition, the topology of a vehicle platoon is sparse and the communication is limited and even unreliable sometimes. Therefore, the decentralized control architecture or the distributed control architecture is preferred to a central coordinator for control of vehicle platoons. Specifically, DeMPC scheme and DMPC scheme will be explored for control of platoons.

In addition, the topology of the whole networked systems can be time-varying due to the joining of new subsystems or the removal of previous subsystems, which introduces a concept of plug and play control [5] [8]. Plug and play capabilities for control involves two important tasks: 1. To avoid complete redesign of controllers for new topology, distributed synthesis of local controllers based on local information is required. 2. The new topology shall be steered to a feasible initial state. Considering that vehicle platoons
will have vehicles that join and leave the existing platoon, the resulting DeMPC and DMPC schemes should have the ability to achieve plug-and-play operation. However, we will simplify the second task by assuming that there are other controllers or protocols to achieve a new feasible state. The distributed synthesis will be our primary focus.

1.1.2 MPC for vehicle platoons

For platoon control using MPC, three important issues related to the controller design are recursive feasibility guarantee, string stability guarantee and the type of platoon model used. Recursive feasibility means that the MPC problem always has a solution if it is initially feasible. String stability describes the ability of a platoon in attenuating disturbances introduced by the leader while moving down stream in the platoon. String stability is important in practice to avoid traffic jams. Regarding the platoon model, different spacing policies can be used [9], leading to different models. Some platoon control algorithms are designed for the so-called constant spacing policy, where the desired distance between two vehicles is constant. Other platoon models use the so-called velocity-dependent spacing policy where the desired distance is a function of the vehicles velocity. It is found that the velocity dependent spacing policy can assure string stability without vehicle-to-vehicle communication, while communication is required for constant spacing policy to guarantee string stability [10].

Some DMPC schemes have been proposed for vehicle platooning control. In [11], a DMPC algorithm was proposed which focused on achieving string stability. However, the guarantee of recursive feasibility was simplified by constraining the predicted terminal state to the origin and only constant spacing policy was considered. Another DMPC algorithm was used and implemented experimentally as shown in [12] [13], which considered string stability and a velocity-dependent spacing policy. However, no guarantee of recursive feasibility was provided therein.

A DeMPC scheme for vehicle platooning control was proposed in [14]. The authors of [14] proposed a DeMPC scheme based on the robust model predictive control (RMPC) algorithm from [1]. The coupled input of a subsystem was regarded as disturbance and a local RMPC controller was employed for each vehicle, which was robust against the influence of its neighbor. However, it was found that the resulting DeMPC could not achieve the convergence to the desired distance. The authors of [14] also made effort to
develop a new DMPC scheme by establishing communication among the local RMPC algorithms. However, recursive feasibility could not be guaranteed by the proposed DMPC scheme. In addition, only constant spacing policy was considered and string stability was also not discussed.

In this work, the focus is to design DeMPC and DMPC schemes for vehicle platooning using velocity dependent policy with important properties of MPC theory, i.e. stability and recursive feasibility. The results are expected to contribute to MPC theory and to demonstrate its potential application in vehicle platooning. Simplification is made by ignoring string stability constraints as proposed in [12] for platoons.

Inspired by [14], we will start with RMPC algorithms to formulate a DeMPC scheme, where no communication is required. Then, DMPC scheme will be obtained by establishing the communication among the local RMPC controllers to reduce the conservativeness, i.e. enlarge the feasible region. In this work, two RMPC algorithms, i.e. [1] [2], will be chosen for DeMPC formulation and we will provide methods for achieving the distribution of these algorithms. The reason why we choose these two RMPC algorithms will be discussed in the preliminaries of Chapter 2. In addition, distributed synthesis of the resulting DeMPC and DMPC schemes will also be briefly discussed in this thesis.

1.2 Outline and Contribution

The organization of the thesis is shown in Figure 1.1.

In Chapter 2, the preliminaries will be introduced, including the system and control theory, the basics of RMPC algorithms such as the notation of open-loop RMPC and feedback RMPC, and the models of vehicle platoon. In the end of Chapter 2, the problems considered in this thesis will be formulated.

Chapter 3 and Chapter 4 consider different RMPC algorithms. In Chapter 3, the open-loop RMPC algorithm in [1] will be presented. Firstly, a DeMPC scheme will be formulated for the platooning control problem based on the open-loop RMPC algorithm. In this part, we provide a proof of stability for the RMPC algorithm which is not addressed in [1]. Then, a DMPC scheme is developed based on the resulting DeMPC formulation and the main goal of this part is to provide the proof of recursive feasibility for the
DMPC problem. After that, the algorithms to achieve plug and play operations for both the DeMPC scheme and the DMPC scheme are provided. In the end of Chapter 3, numerical analysis is discussed based on the simulation results.

Chapter 4 starts from the feedback RMPC algorithm in [2]. With the similar structure as Chapter 3, we will first formulate the DeMPC scheme based on the feedback RMPC algorithm, which is followed by the new DMPC scheme and the proof of recursive feasibility for it. After that, distributed synthesis will also be briefly mentioned. Finally, numerical analysis of the DeMPC and the DMPC formulations is provided. Most importantly, in the end of Chapter 4, the DeMPC and DMPC scheme of Chapter 4 will be compared with the ones of Chapter 3.

In Chapter 5, some application issues of vehicle platooning will be discussed. Especially, string stability will also be analyzed. The thesis is then finalized by some conclusions and recommendations in Chapter 6.
Chapter 2

Preliminaries and Problem formulation

In the preliminaries, we will first introduce the basics of system and control theory, including the formulation of general constrained linear systems with additive disturbance and the definitions of stability for constrained systems. Then, the basic features of the two RMPC algorithms, i.e. [1] [2], will be explained, which can facilitate the understanding of the following chapters. In the end of the preliminaries, the models of vehicle platoon will be introduced.

Finally, the problems to be solved in this work will be formulated.

2.1 Preliminaries

2.1.1 Mathematical preliminaries

**Definition 2.1.** Given two sets $\Omega \subset \mathbb{R}^n$ and $\Phi \subset \mathbb{R}^n$, the Pontryagin difference between $\Omega$ and $\Phi$ is defined as

$$\Omega \sim \Phi \triangleq \{x \in \mathbb{R}^n | x + y \in \Omega, \forall y \in \Phi\}.$$  \hspace{1cm} (2.1)
Definition 2.2. Given two sets $\Omega \subset \mathbb{R}^n$ and $\Phi \subset \mathbb{R}^n$, the Minkowski sum of $\Omega$ and $\Phi$ is defined as

$$\Omega \oplus \Phi \triangleq \{ z \in \mathbb{R}^n | \forall x \in \Omega, y \in \Phi : z = x + y \}.$$ (2.2)

A block-diagonal matrix is denoted by $diag(C_1, \ldots, C_n)$ with matrices $C_1$ to $C_n$ on the main diagonal and zeros everywhere else. For a vector $x \in \mathbb{R}^n$, let $||x||_P$ denote the quadratic form of $x$, i.e. $||x||_P = x^T P x$.

2.1.2 System and control theory

System formulation

We consider a discrete-time linear system with additive disturbance described by

$$x_{k+1} = Ax_k + Bu_k + w_k, \quad k \in \mathbb{N}$$ (2.3)

which is subject to the following constraints:

$$x_k \in \mathcal{X} \subset \mathbb{R}^n, \quad u_k \in \mathcal{U} \subset \mathbb{R}^m, \quad \forall k \in \mathbb{N},$$ (2.4)

where $x_k$ is the state, $u_k$ is the control input and $w_k \in \mathcal{W} \subset \mathbb{R}^n$ is an unknown disturbance. The sets $\mathcal{U}$, $\mathcal{X}$ and $\mathcal{W}$ are all convex, compact and contain the origin as an interior point. In addition, it is assumed that state feedback $x_k$ can be measured at every time instant.

Consider that system (2.3) is controlled by a certain control law $u_k = \kappa(x_k)$, the closed-loop system is given by

$$x_{k+1} = Ax_k + B\kappa(x_k) + w_k.$$ (2.5)

In addition, the corresponding nominal of system (2.3) can be formulated as follows:

$$\bar{x}_{k+1} = A\bar{x}_k + B\bar{u}_k$$ (2.6)
If system (2.6) is controlled by a control law $u_k = \kappa(x_k)$, the closed-loop nominal system is given by

$$\bar{x}_{k+1} = A\bar{x}_k + B\kappa(\bar{x}_k) \quad (2.7)$$

Then, we denote the solution of equation (2.5) at sampling time $k$, for initial state $x_0$ and the disturbance sequence $w$ as $\phi_\kappa(k, x_0, w)$, where $w = \{w_0, ..., w_{k-1}\}$. Similarly, the solution of the closed-loop nominal system (2.7) is denoted by $\bar{\phi}_\kappa(k, x_0) \triangleq \phi_\kappa(k, x_0, 0)$.

**Assumption 2.1.** The pair $(A, B)$ is stabilizable.

**Stability for constrained systems**

In the closed-loop system (2.5), providing stability guarantee will be the main goal of the controller. Due to the fact that system (2.5) also has constraints, stability will not be stated globally but defined in a local feasible region. Therefore, notion of invariant sets will be first introduced, which is followed by local stability.

**a. Invariant set definitions**

**Definition 2.3.** Given a set $\Omega \subset \mathbb{R}^n$, the robust one-step set $\tilde{Q}(\Omega)$ for system (2.3) and the one-step set $Q(\Omega)$ for system (2.6) is defined as

$$\tilde{Q}(\Omega) \triangleq \{x_k \in \mathbb{R}^n | \exists u_k \in U : Ax_k + Bu_k + w_k \in \Omega, \forall w_k \in \mathbb{W}\},$$

$$Q(\Omega) \triangleq \{\bar{x}_k \in \mathbb{R}^n | \exists \bar{u}_k \in U : A\bar{x}_k + B\bar{u}_k \in \Omega\}.$$  

**Remark 2.1.** [1] Given a set $\Omega \subset \mathbb{R}^n$, based on the definition of the Pontryagin difference and Definition 2.3, the following relationship holds for system (2.3):

$$\tilde{Q}(\Omega) = Q(\Omega \sim \mathbb{W}).$$

**Remark 2.2.** For all $\Omega_1, \Omega_2$,

$$\Omega_1 \subseteq \Omega_2 \Rightarrow Q(\Omega_1) \subseteq Q(\Omega_2).$$

**Definition 2.4.** Given a control law $u_k = \kappa(x_k)$, the set $\Phi \subseteq \mathbb{X}$ is a robust positively invariant (RPI) set for the closed-loop system $x_{k+1} = Ax_k + B\kappa(x_k) + w_k$ if and only if $\forall k \in \mathbb{N}$ and $\forall x_k \in \Phi$, it holds that $x_{k+1} \in \Phi$ and $u_k \in U$, $\forall w_k \in \mathbb{W}$. 
**Definition 2.5.** The set $\Phi \subseteq X$ is a robust control invariant (RCI) set for the subsystem $x_{k+1} = Ax_k + Bu_k + w_k$ if and only if $\forall k \in \mathbb{N}, \forall x_k \in \Phi, \exists u_k \in \mathbb{U}$ such that $x_{k+1} \in \Phi$, $\forall w_k \in \mathbb{W}$.

**Theorem 2.6.** [1] The set $\Phi \subset \mathbb{R}^n$ is a robust control invariant set if and only if $\Phi \subseteq \tilde{Q}(\Phi)$.

**Remark 2.3.** Given system (2.3) or closed-loop system (2.5), the state constraint and the input constraint, the existence of RCI set or RPI set rely on the size of the disturbance set. If the size of the disturbance set is too large, the RCI set or the RPI set may not exist.

It is easy to extend the definitions of RPI set and RCI to the nominal closed-loop system (2.7) and the nominal system (2.6) where the disturbance equals to zero, which leads to positively invariant (PI) set and control invariant (CI) set, respectively.

The concept of invariant sets implies that the state trajectory always stay in certain set if the initial state starts in the set. This property is important for control problems with constraints. Considering that the invariant set is also constraint admissible, the states will then always satisfy the constraints if the initial state belong to the set.

Now, if we can find a region where the constraints can always be satisfied, local stability can be defined in this region.

**b. Stability definitions**

We will first define stability for nominal closed-loop system (2.7) in some PI set $\Phi$. However, in the presence of disturbance, the definition of stability shall be extended to the perturbed closed-loop system (2.5), which leads to the concept of input-to-state stability (ISS).

**Definition 2.7.** Given a PI set $\Phi$ including the origin as an interior-point, the closed-loop nominal system $\bar{x}_{k+1} = A\bar{x}_k + B\kappa(\bar{x}_k)$ is asymptotically stable in $\Phi$ if $\forall x_0 \in \Phi$ there exists $KL$-function $\beta$ such that

$$
\|\tilde{\phi}_\kappa(k, x_0)\| \leq \beta(\|x_0\|, k), \quad \forall k \in \mathbb{N}.
$$

(2.9)
This property is often demonstrated by the existence of a Lyapunov function. To extend the concept of stability to the perturbed system (2.5), the following definition of ISS is used:

**Definition 2.8.** Given a set $\Phi \subseteq \mathbb{R}^n$ including the origin as an interior-point, the closed-loop system $x_{k+1} = Ax_k + B\kappa(x_k) + w_k$ is input-to-state stable (ISS) in $\Phi$ with respect to $w_k \in \mathcal{W}$ if there exists a $\mathcal{KL}$-function $\beta$ and a $\mathcal{K}$-function $\gamma$ such that

$$
\|\phi_\kappa(k, x_0, w)\| \leq \beta(\|x_0\|, k) + \gamma(\|w_{0:k-1}\|), \quad \forall k \in \mathbb{N}
$$

(2.10)

for all initial state $x_0 \in \Phi$ and all disturbance sequence $w \triangleq \{w_0, ..., w_{k-1}\}$ where $w_l \in \mathcal{W}$ for all $l \in \{0, ..., k-1\}$. In addition, $\|w_{0:k-1}\| \triangleq \max_{0 \leq j \leq k-1} \|w_j\|$.

Note that ISS implies the origin is an asymptotically stable point for the nominal model (2.7).

### 2.1.3 Basics of robust MPC

In this subsection, we will introduce the basics of the two robust MPC algorithms from [1] [2] chosen in this work.

The two RMPC problems will first be formulated for system (2.3). After the two different formulations are obtained, the goal is to explain the three basic features of general RMPC algorithms based on the two formulations and compare these features of the two RMPC algorithms. However, the important theoretical proof for the two RMPC problems, such as ISS and recursive feasibility, will only be addressed in detail in Chapter 3 and Chapter 4. The reason why the proof is separate from preliminaries is because the three basic features are general for all RMPC algorithms, by contrast, the proof of ISS and recursive feasibility depends on the specific RMPC scheme.

First of all, the two different RMPC problems are formulated as follows:

**Open-loop RMPC**

The open-loop RMPC problem in [1] can be formulated for system (2.3) as follows:
Problem 2.1.

\[
\min_{\mathbf{u}_k} ||\mathbf{x}_{N|k}|| + \sum_{l=0}^{N-1} ||\mathbf{x}_{l|k}|| Q_x + ||\mathbf{u}_{l|k}|| Q_u
\]

s.t. \(\mathbf{x}_{l+1|k} = A\mathbf{x}_{l|k} + B\mathbf{u}_{l|k}, \quad \mathbf{x}_{0|k} = x_k,\)
\(\mathbf{x}_{l|k} \in \mathcal{X}, \quad \mathbf{u}_{l|k} \in \mathcal{U}, \quad \forall l \in \{0, ..., N-1\},\)
\(\mathbf{x}_{N|k} \in \mathcal{T}, \quad \mathbf{x}_{1|k} \in \mathcal{X}_R \sim \mathcal{W}, \quad \mathbf{u}_k = \mathbf{u}_{0|k},\)

where \(\mathbf{u}_k\) denotes the vector of predicted input sequence \(\{u_{0|k}^T, ..., u_{N-1|k}^T\}\), \(N\) is the prediction horizon and \(\mathcal{T}\) is the terminal set. \(\mathcal{X}_R\) is a set to be designed such that the constraint \(\mathbf{x}_{1|k} \in \mathcal{X}_R \sim \mathcal{W}\) can ensure ISS. The matrices \(P, Q_x\) and \(Q_u\) are the cost matrices used to penalize the state and the input deviation from the origin along the prediction horizon. How to choose these parameters to ensure ISS and recursive feasibility will be discussed in Chapter 3.

Feedback RMPC

The feedback RMPC problem in [2] can be formulated for system (2.3) as follows:

Problem 2.2.

\[
\min_{\mathbf{M}_k, \mathbf{v}_k} ||\mathbf{x}_{N|k}|| + \sum_{l=0}^{N-1} ||\mathbf{x}_{l|k}|| Q_x + ||\mathbf{u}_{l|k}|| Q_u
\]

s.t. \(\mathbf{x}_{l+1|k} = A\mathbf{x}_{l|k} + B\mathbf{u}_{l|k} + \mathbf{w}_{l|k}, \quad \mathbf{x}_{0|k} = x_k\)
\(\mathbf{u}_{l|k} = \sum_{j=0}^{l-1} M_{(l,j)|k} w_{j|k} + v_{l|k}, \quad \mathbf{u}_k = \mathbf{u}_{0|k},\)
\(\mathbf{x}_{l|k} \in \mathcal{X}, \quad \mathbf{u}_{l|k} \in \mathcal{U}, \quad \forall \mathbf{w}_{l|k} \in \mathcal{W}, \quad \forall l \in \{0, ..., N-1\}\)
\(\mathbf{x}_{N|k} \in \mathcal{T},\)

where \(M_{(0,0)|k}\) and \(M_{(0,-1)|k}\) are matrices with all entries equal to zero. In addition, the input sequence is parameterized as a function of the disturbance sequence, i.e. \(\mathbf{u}_k = \mathbf{u}_{0|k} + \sum_{j=0}^{l-1} M_{(l,j)|k} w_{j|k} + v_{l|k},\)
Problem formulation

\[ M_k w_k + v_k, \text{ where } w_k = [w_{0[k]}^T, \ldots, w_{N-1[k]}^T]^T \text{ and } M_k, v_k \text{ are defined as} \]

\[
M_k \triangleq \begin{bmatrix}
0 & \ldots & \ldots & 0 \\
M_{(1,0)[k]} & 0 & \ldots & 0 \\
\vdots & \ddots & \ddots & \vdots \\
M_{(N-1,0)[k]} & \ldots & M_{(N-1,N-2)[k]} & 0
\end{bmatrix}, \quad v_k \triangleq \begin{bmatrix}
v_{0[k]} \\
\vdots \\
v_{N-1[k]}
\end{bmatrix}.
\]

(2.11)

Also, the way to choose all the parameters of Problem 2.2 to ensure ISS and recursive feasibility will be discussed in detail in Chapter 4.

Remark 2.4. Instead of \( \bar{u} \) in model (2.6), \( u \) is used in Problem 2.1 and Problem 2.2. This is because the control input \( u \) is calculated by the two problems based on the feedback of real state \( x \) instead of the nominal state \( \bar{x} \).

Three features of the two RMPC problems

The three features of general RMPC algorithms include:

- The type of prediction: Open-loop prediction or Feedback prediction;
- The type of model used in constraints: nominal model (2.6) or perturbed model (2.3);
- The type of model used in cost function: nominal model (2.6) or perturbed model (2.3).

For the type of prediction, we say that Problem 2.1 uses open-loop prediction and Problem 2.2 uses feedback prediction, which is why we call the first algorithm open-loop RMPC and the second algorithm feedback RMPC. The difference in prediction type can be seen from if the predicted optimal input sequence depends on the disturbance sequence over the prediction horizon.

Assuming that there is a feedback state \( x_k \) and two different possible disturbance sequences, including \( w_k^1 = [w_{0[k]}^T, \ldots, w_{N-1[k]}^T]^T \) and \( w_k^2 \). In Problem 2.1, because the \( u_k \) is the optimization variable, the optimal input sequence \( u_k^* = [u_{0[k]}^T, \ldots, u_{N-1[k]}^T]^T \) will not change based on two different disturbance sequence but depends only on the feedback state \( x_k \). Thus, we call this type of prediction the open-loop prediction.
However, in Problem 2.2, the optimization variables are $M_k$ and $v_k$. With the obtained optimal solution $M_k^*$ and $v_k^*$, two optimal input sequences will be obtained based on the two disturbance sequences, including $u_{k}^{*1} = M_k^* w_k^1 + v_k^*$ and $u_{k}^{*2} = M_k^* w_k^2 + v_k^*$. Thus, the predicted optimal input sequence $u_k^*$ in Problem 2.2 depends on both the state feedback $x_k$ and the disturbance path $w_k$. We call this type of prediction the (disturbance) feedback prediction.

It can be seen that given a feedback state $x_k$, Problem 2.1 tends to use a fixed optimal input sequence $u_k^*$ to compensate all possible disturbance sequences $w_k$. However, Problem 2.2 uses a set of input sequences to compensate $w_k$. Thus, we can expect that the feedback prediction tends to give Problem 2.2 a larger feasible region than the one of Problem 2.1.

As for the type of model used in constraints, it can be seen that Problem 2.1 uses nominal model (2.6) and Problem 2.2 uses perturbed model (2.3). The advantage of using the perturbed model in the constraints is that a good prediction can be achieved because real state is considered in the whole prediction horizon. The disadvantage of using perturbed model is that it can introduce more conservativeness than the the problems using the nominal model in their constraints. This is due to the fact the disturbance have to be compensated on the whole horizon. Thus, it can be expected that the usage of the perturbed model in the constraints tends to give a smaller feasible region for Problem 2.2 than the feasible region of Problem 2.1.

For the type of model used in the cost function, both Problem 2.1 and Problem 2.2 use nominal model (2.6). The advantage of using nominal model in cost function is that it avoids the min-max optimization as shown in [15], which reduces the computational effort. The disadvantage is that the predicted cost is not accurate because real state is not used, which may not lead to the minimal worst cost of the real system.

Overall, based on the first two features, i.e. the prediction type and the model type in constraints, Problem 2.2 tends to gain larger feasible region from its feedback prediction but also tends to obtain smaller feasible region from the usage of perturbed model in the constraints. Thus, for now, it is hard to make a conclusion that which formulation will have a large feasible region. Instead, the feasible regions of the two RMPC algorithms will be compared for the specific platooning control problem in Chapter 4.
After having a understanding of the basic features of the RMPC algorithms, we can explain the reasons why the two robust controllers are chosen. For the open-loop RMPC Problem 2.1, it is chosen because of its simplicity, i.e. only the nominal model is used in both its constraints and cost function. For the feedback RMPC Problem 2.2, it is chosen because it has feedback prediction and has no min-max optimization. The feedback prediction might reduce conservativeness caused by the open-loop prediction of Problem 2.1 and no min-max optimization makes it consume less computational effort than the effort required by the traditional min-max robust MPC schemes.

Finally, after the basics of system and control theory and robust MPC, the models of vehicle platoon will introduced next.

2.1.4 Working mechanism of platoons

The longitudinal vehicle platoon consists of a group of vehicles in a chain structure where the leader tracks a reference and the followers should achieve automatic following while keeping a safe distance at all times. In this section, we will introduce the models of the platoon.

Even if the vehicles are physically decoupled, the models of the subsystems in a platoon are coupled through inputs because the models describe the change of the relative distance and the relative speed between two vehicles. Due to the coupling, we call these models distributed models which will be formulated first in this section.

Then, when the communication among subsystems is not available, the decentralized models will also be formulated by regarding the input coupling as disturbance which is unknown but belongs to a set. In this case, the models are called decentralized models because no coupling appears in the equations.

Distributed models

It is assumed that there are $N_A$ vehicles in a platoon and the leader is denoted by vehicle 1. Considering two adjacent vehicles in Figure 2.1, let $p^i$ denote the position and $v^i$ represent the velocity of vehicle $i$ respectively, where $i \in \{1, \ldots, N_A\}$. The desired distance between the two vehicles is $d_s + hv^i$, where $d_s$ is the desired distance when
vehicle $i$ has zero velocity and $h$ is the headway time. The headway time is a designed value which is fixed and represents the time that the vehicle takes to reach its preceding vehicle at current speed. The constant spacing policy is when $h = 0$ such that the desired distance is $d_s$ and time-invariant. By contrast, the velocity-dependent spacing policy is when $h > 0$.

The states of the inter-vehicle dynamics in the MPC problem are formulated as follows:

$$x^i = \begin{bmatrix} e^i_p \\ e^i_v \end{bmatrix} = \begin{bmatrix} p^{i-1} - p^i - d_s - hv^i \\ v^{i-1} - v^i \end{bmatrix}, \quad \forall i \in \{2, \ldots, N_A\}, \quad (2.12)$$

where $e^i_p$ represents the relative speed while $e^i_v$ denotes the error between the inter-vehicle distance and the desired distance. The units of $e^i_p$ and $e^i_v$ are meter and meter per second respectively. Note that the state $x^i$ denotes the relative information between two vehicles. The state which contains the relative information between vehicle 1 and vehicle 2 is denoted by $x^2$ because the resulting control input $\kappa^2(x^2)$ is the input of vehicle 2, which will be shown in equation (2.13).

It is assumed that every vehicle is equipped with sensors which can measure the relative distance and the relative velocity $e^i_p$ to its preceding vehicle, which is practical because the information can also be obtained by commercially available vehicles with Adaptive Cruise Control (ACC). Thus, the state feedback can be obtained.

Consequently, the discrete-time model of subsystem $i$ can be written as

$$x^i_{k+1} = A^i x^i_k + B^i u^i_k + E^i u^{i-1}_k, \quad \forall i \in \{2, \ldots, N_A\}, \quad (2.13)$$
where $u^i_k$ is the control input which denotes the desired acceleration of vehicle $i$ at time $k$, $u^{i-1}_k$ is external signal which represents the acceleration of vehicle $i-1$ at time $k$ and

$$A^i = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}, \quad B^i = \begin{bmatrix} -hT - T^2/2 \\ -T \end{bmatrix}, \quad E^i = \begin{bmatrix} T^2/2 \\ T \end{bmatrix}, \quad (2.14)$$

where $T$ is the sampling time. As shown in model (2.13), the platoon is called distributed system because every subsystem (2.13) is coupled through the input of its preceding subsystem.

**Remark 2.5.** Another way to understand the model (2.13) is to represent every vehicle by a double integrator which describes the evolution of the absolute position and the absolute speed of the vehicle. Then, to obtain the relative distance and the relative speed between two vehicles, model (2.13) can be regarded as the subtraction between two double integrators while the desired distance is also considered in the subtraction.

The state of subsystem $i$ and the desired acceleration of each vehicle are subject to local constraints

$$x^i \in \mathcal{X}^i, \quad \forall i \in \{2, ..., N_A\}, \quad (2.15a)$$

$$u^n \in \mathcal{U}^n, \quad \forall n \in \{1, ..., N_A\}, \quad (2.15b)$$

where the sets $\mathcal{X}^i$ and $\mathcal{U}^n$ are assumed to be convex, compact and contain the origin in the interior. Note that the leader also respects an input bound, i.e. $\mathcal{U}^1$.

**Remark 2.6.** The leader is expected to track a specified reference within an input bound $\mathcal{U}^1$, which could be achieved by another controller or a driver. To achieve automatic following with safe distance, only the followers are equipped with the designed controllers which calculate the desired acceleration as their inputs.

**Assumption 2.2.** There exists $K^i$ such that $(A^i + B^i K^i)$ is Schur for all $i \in \{2, ..., N_A\}$.

**Remark 2.7.** For simplicity, the dynamic models of all subsystems are chosen to be identical. However, in the proposed algorithm, no assumption on the identical dynamics is made and thus the method can be extended to the platoon with heterogeneous dynamics and general networked systems in a chain structure.

As for simulation, double integrators are used to simulate the vehicles.
Decentralized models

As shown in equation (2.13), the subsystems are coupled through inputs. However, if communication among subsystems is not available, the external signal $u_{i-1}^k$ cannot be known for model (2.17). Thus, the coupling term $E_i u_{i-1}^k$ in equation (2.13) can be regarded as an additive disturbance, i.e.,

$$w_k^i = E_i u_{i-1}^k,$$  

(2.16)

where $w_k^i \in \mathbb{D}^i$, $\forall k \in \mathbb{N}$, and the disturbance set $\mathbb{D}^i$ is defined by $\mathbb{D}^i \triangleq E^i U_{i-1}$. The subsystem (2.13) are then reformulated as

$$x_{k+1}^i = A_i x_k^i + B_i u_k^i + w_k^i, \quad \forall i \in \{2, \ldots, N_A\}.$$  

(2.17)

Note that the coupling of subsystems is removed in every subsystem (2.17) and thus the overall platoon is fully decentralized. For each subsystem (2.17), the disturbance is not pre-known but belongs to the set $\mathbb{D}^i$.

In addition, we define the nominal model for subsystem $i$ as

$$\bar{x}_{k+1}^i = A_i \bar{x}_k^i + B_i \bar{u}_k^i$$  

(2.18)

where $\bar{x}_k^i$ represents the nominal state of subsystems $i$ at time $k$, $\forall k \in \mathbb{N}$ and $\forall i \in \{2, \ldots, N_A\}$. Note in the nominal model (2.18), we did not use nominal input $\bar{u}_k^i$. This is because that the control input calculated by the MPC law is always a function of the real state $x_k^i$.

2.2 Problem formulation

There are three problems that need to be solved, including the decentralized control problem, the distributed control problem and the distributed synthesis for plug and play operations.
2.2.1 Decentralized control problem

First of all, the control problem is that given $N_A$ vehicles with the model shown in equation (2.13), for all $i \in \{2, \ldots, N_A\}$, synthesize the control input $u_k^i$ such that $\lim_{k \to \infty} x_k^i = 0$ and $x_k^i \in X^i$, $u_k^i \in U^i$ are satisfied all the time. For a vehicle platoon, $x_k^i = 0$ means that the relative velocity between two vehicles is zero and the relative distance equal to the desired distance.

As for decentralized control problem, the goal is to achieve the control problem in a decentralized way, i.e. using decentralized model (2.17) and without any communication among subsystems. Motivated by the model (2.17), the two robust MPC controllers from [1] and [2] will be exploited to solve the decentralized control problem. In addition, the guarantee of ISS and recursive feasibility should be provided.

The DeMPC scheme based on the open-loop RMPC will be named as open-loop DeMPC. By contrast, the DeMPC based on the feedback RMPC will be named as feedback DeMPC.

2.2.2 Distributed control problem

Since the input coupling $E^iu^{i-1}$ is regarded as disturbance which is bounded by $D^i$, the robust controllers used for decentralized control problem will compensate all possible values of $E^iu^{i-1}$ belonging to $D^i$. However, if the information related to the value of $u^{i-1}$ can be communicated, the feasible region can be larger because the value of disturbance is known. This control scheme leads to a distributed MPC control scheme because the communication among subsystems is required.

Thus, the distributed control problem is to achieve the control problem in a distributed way, i.e. using decentralized model (2.17) and with communication among subsystems. In addition, ISS and recursive feasibility should still hold after the communication is established.

The DMPC scheme based on the open-loop RMPC Problem 2.1 will be named as open-loop DMPC. By contrast, the DMPC based on the feedback RMPC Problem 2.2 will be named as feedback DMPC.
2.2.3 Distributed synthesis of plug and play operations

The third problem is to achieve the plug and play operations, i.e. new vehicles join an existing platoon or vehicles leave a platoon while stability and recursive feasibility still hold.

Two actions, including the automatic synthesis of the local controllers and an initialization process to steer the new platoon to a feasible initial position, can be applied to achieve plug and play operations. Only after the initialization, the new controllers can be applied to the resulting new platoon with feasible initial states. In this work, we simplify the initialization process since it requires complex protocols e.g. merging protocols in [16]. The focus will lie on automatizing the distributed synthesis of the local controllers of DeMPC and DMPC schemes when the topology of the platoon changes.
Chapter 3

Open-loop DeMPC and DMPC

In the first part of this chapter, the solution to ensure the existence of RCI and RPI sets for robust controller synthesis is proposed. This is the first step towards the solution of the control problem. Note that this method is independent on the MPC scheme and will be used in both Chapter 3 and Chapter 4.

Then, the problems formulated in Section 2.2 will be solved based on the two robust MPC controllers from [1] and [2] in Chapter 3 and Chapter 4, respectively. In this part, both chapters follow the same structure.

In each chapter, we will first discuss the decentralized design using robust MPC controllers for the control problem of vehicle platoons, i.e. the robust controllers are applied to platoons directly by regarding the input coupling shown in equation (2.13) as disturbance and no communication during operation is needed in this case.

Then, the DMPC algorithm will be designed based on the decentralized architecture by establishing communication among local controllers. Especially, the conditions of the distribution process under which recursive feasibility is still preserved for the resulting DMPC algorithm will be discussed. This is followed by distributed synthesis of local controllers of DeMPC and DMPC schemes during Plug and Play operations.

Finally, in the end of each chapter, simulation results are presented, which demonstrates that both decentralized control architecture and distributed architecture can achieve the control goal. In addition, we compare the resulting performance of the DMPC algorithm and it of the corresponding DeMPC algorithm.
3.1 Input scaling

Before applying the RMPC controllers, because the RPI and RCI sets are normally required for robust MPC controllers, we will first discuss how to ensure the existence of RPI and RCI sets. Only after having these sets, the robust controllers can be formulated.

The existence of the RCI set and RPI set is a complex issue which is based on the model (2.17), the state constraint $X_i$, the input constraint $U_i$ and the disturbance set $D_i$ which is defined by $E_i^i U_i$. Any of these factors can cause the non-existence of these sets. A sufficient condition to guarantee the existence of the robust control invariant set is out of the scope of this thesis. However, given a model, a state constraint and an input constraint based on physical properties of the system and the safety reasons, the only parameter that we can tune is the size of the disturbance set $E_i^i U_i$ as shown in [14]. The idea is to tune the disturbance set such that it is sufficiently small and then RPI/RCI sets exist. The authors of [14] obtain the input bounds for all vehicles as follows:

$$U_i^{-1} = cU_i, \quad \forall i \in \{2, \ldots, N_A\},$$  \hspace{1cm} (3.1)

where $0 < c \leq 1$ and the input bound of the last vehicle $U^{N_A}$ is chosen as the physical limit of its input. This indicates that the follower always has a larger or equal range of inputs to compensate the behaviour of its preceding vehicle. To achieve a better tuning, we using a scaling matrix $C$ instead of the scalar $c$. Note that the input constraint set $U_i$ is defined as

$$U_i \triangleq \{u_i \in \mathbb{R} | u_{i_{\min}} \leq u_i \leq u_{i_{\max}}\}, \quad \forall i \in \{1, \ldots, N_A\},$$ \hspace{1cm} (3.2)

thus, a line vector $U_i \triangleq [u_{i_{\min}}, u_{i_{\max}}]$ can be used to characterize $U_i$. Then the input bounds can be scaled by the scaling matrix $C$ as follows:

$$U_i^{-1} = [u_i^{-1}_{i_{\min}}, u_i^{-1}_{i_{\max}}] = U_i C, \quad \forall i \in \{2, \ldots, N_A\},$$ \hspace{1cm} (3.3)

where $C$ is a diagonal matrix whose diagonal entries can be tuned to obtain desired $U_i^{-1}$, and $U^{N_A}$ is taken as the physical bound of the input for the last vehicle.

For example, assuming that in a three-vehicle platoon, the physical input bound of the last vehicle is $U^3 = [-5, 3]$ and the scaling matrix is $C = \text{diag}(0.9, 0.9)$, then the
input bounds for the first two vehicles are $U^2 = [-5, 3]C = [-4.5, 2.7]$ and $U^1 = [-5, 3]C^2 = [-4, 2.4]$.

**Assumption 3.1.** Given the model, the state constraint $X^i$ and the input constraint $U^i$, it is assumed that there exists a diagonal matrix $C = \text{diag}(c_1, c_2)$ where $0 < c_1, c_2 \leq 1$ such that under the resulting $U^{i-1}$ obtained using equation (3.3), the RCI set for system (2.17) or the RPI set for closed-loop system (2.17) with a control law $u^i_k = K^i x^i_k$ exists.

### 3.2 Decentralized control with RMPC

Using the RMPC algorithm from [1] based on model (2.17), a local MPC problem can be formulated for subsystem $i$ at time $k$, $\forall i \in \{2, ..., N_A\}$ and $\forall k \in \mathbb{N}$, as follows:

**Problem 3.1.**

$$
\min_{u^i_k} ||\bar{x}^i_{N^i|k}||_{P_x} + \sum_{l=0}^{N^i-1} ||\bar{x}^i_{l|k}||_{Q_x} + ||u^i_{l|k}||_{Q_u}
$$

s.t.  
- $\bar{x}^i_{l+1|k} = A^i \bar{x}^i_{l|k} + B^i u^i_{l|k}$,  
- $\bar{x}^i_{0|k} = x^i_k$,  
- $\bar{x}^i_{l|k} \in X^i$,  
- $u^i_{l|k} \in U^i$,  
- $\forall l \in \{0, ..., N^i - 1\}$,  
- $\bar{x}^i_{N^i|k} \in T^i$,  
- $\bar{x}^i_{1|k} \in X_R^i \sim D^i$,  
- $u^i_k = u^i_{0|k}$,

where $\bar{x}^i_{l|k} \in X_R^i \sim D^i$ is a robustness constraint and $X_R^i$ is chosen to be the maximal robust control invariant set of the subsystem $i$. $T^i$ is the terminal set and $u^i_k$ denotes the input sequence $\{u^i_{0|k}, ..., u^i_{N^i-1|k}\}$ where $N^i$ is the prediction horizon of the local optimization problem. $P^i$, $Q_x^i$ and $Q_u^i$ are chosen cost matrices for the predicted terminal state, the state sequence and the input sequence, respectively.

By initialization, every following vehicle is assigned with a local Problem 3.1 with the local parameters. The leading vehicle will be fed with an input reference within an input bound $U^1$ which is calculated using equation (3.3). The way to choose the local parameters of Problem 3.1 will be discussed later. After initialization, for all $k \in \mathbb{N}$ and $i \in \{2, ..., N_A\}$, the leader generates its input based on the reference while the followers solve local Problem 3.1 in parallel and apply $u^i_{0|k}$ as the control action $u^i_k$. 
3.2.1 Stability and Recursive feasibility

The parameters of Problem 3.1 should be chosen such that ISS can be guaranteed. Thus, conditions are provided to guarantee recursive feasibility of Problem 3.1 and ISS of the closed-loop system under the MPC control law.

**Definition 3.1.** For Problem 3.1 without constraint \( \bar{x}^i_{1|k} \in X^i_{R} \sim \mathbb{D}^i \), the feasible set \( X^i(T^i, N^i) \) is defined as

\[
X^i(T^i, N^i) \equiv \{ x^i_k \in \mathbb{R}^n | \exists u^i_k : (x^i_k, u^i_k) \text{satisfies all constraints except that } \bar{x}^i_{1|k} \in X^i_{R} \sim \mathbb{D}^i \}.
\] (3.4)

Note that \( X^i(T^i, N^i) = Q^i[X^i(T^i, N^i-1)] \cap \mathcal{X}^i \), where \( Q^i \) is the one-step set for subsystem \( i \). In what follows, we will make use of the following alternative characterization of the feasible set \( X^i_F(T^i, N^i - 1) \) with respect to Problem 3.1 with horizon \( N^i \): in this case, \( X^i_F(T^i, N^i - 1) \) is the set of (predicted) states \( \bar{x}^i_{1|k} \) for which the input sequence \( \{ u^i_{1|k}, \ldots, u^i_{N^i - 1|k} \} \) exists such that the constraints of Problem 3.1 except \( \bar{x}^i_{1|k} \in X^i_{R} \sim \mathbb{D}^i \) are satisfied.

**Theorem 3.2.** [1] For any \( i \in \{2, \ldots, N_A\} \), if \( X^i_{R} \) is a robust control invariant set of the subsystem \( i \) within the distributed systems (2.17) and

\[
X^i_{R} \sim \mathbb{D}^i \subseteq X^i_F(T^i, N^i - 1),
\] (3.5)

then the corresponding local MPC Problem 3.1 is recursively feasible.

Based on Theorem 3.2, given \( X^i_{R} \) and \( \mathbb{D}^i \), the prediction horizon \( N^i \) can be tuned such that recursive feasibility is achieved.

As reported in [14], the convergence of the states to the origin can not always be achieved because the set \( X^i_{R} \sim \mathbb{D}^i \) does not always contain the origin. To solve this issue, we make use of following remark:

**Remark 3.1.** Given two sets \( \Omega \subset \mathbb{R}^n \) and \( \Phi \subset \mathbb{R}^n \), \( \Omega \sim \Phi \) contains the origin if and only if \( \Phi \subseteq \Omega \).

However, note that no conclusion on the condition under which the origin is an interior point of the Pongryagin difference is made. Remark 3.1 only shows that the entries of
the scaling matrix $C$ described in Section 3.1 may have upper bounds and they should be chosen such that $X^i_R \sim D^i$ contains the origin in its interior. To enlarge $U^{i-1}$ such that vehicle $i-1$ has a looser input bound, $C$ should have the entries with largest values while the origin is an interior point of the Pontryagin difference.

The conditions for stability can be collected in the following theorem:

**Theorem 3.3.** If the following conditions hold:

1. Conditions of Theorem 3.2 are satisfied and $X^i_R \sim D^i$ contains the origin in its interior,
2. The cost matrices $Q^i_x$ and $Q^i_u$ are positive definite,
3. The terminal set $T^i$ is a PI set for the nominal system (2.18) under a local stabilizing control law $u^i_k = K^i x^i_k$ and $T^i$ is constraint admissible,
4. Terminal cost matrix $P^i$ satisfies that $P^i \succ 0$ and

\[
(A^i + B^i K^i)^T P^i (A^i + B^i K^i) - P^i \preceq -Q^i_x - K^i Q^i_u K^i,
\]  

then, the perturbed system (2.17) in closed-loop with the MPC control law of Problem 3.1 is ISS in the feasible region of Problem 3.1 with respect to $w^i \in D^i$.

**Proof.** It is well known that under conditions 2 to 4, the optimal cost function is a Lyapunov function for the the nominal system (2.18) in closed-loop with the MPC control law of Problem 3.1. In addition, based on Theorem 1 in [17], under condition 2, the optimal cost function, i.e. the Lyapunov function, is continuous in the feasible region. And based on Theorem 3.3.4 in [18], considering that the feasible region of Problem 3.1 is compact because all constraints in Problem 3.1 are compact, then the Lyapunov function is uniformly continuous in the feasible region. Furthermore, based on condition 1, we know that the feasible region of Problem 3.1 is a RPI set for the perturbed system (2.17) under the MPC control law, which ensures recursive feasibility.

Finally, based on Theorem 4.15 in [8], since there exist a Lyapunov function which is uniformly continuous for the nominal closed-loop system in the feasible region which is also a RPI set, the perturbed system (2.17) under the MPC control law is ISS in the feasible region of Problem 3.1 with respect to $w^i \in D^i$. 
3.3 Distributed control with DMPC

The set \( \mathcal{X}_k \sim \mathbb{D}^i \) is conservative because it compensates all possible values of the disturbance belonging to the set \( \mathbb{D}^i \). Due to the fact that the disturbance is defined as \( w_k^i = E^i u_k^{i-1} \), the value of the disturbance can be known in advance if the input \( u_k^{i-1} \) of the preceding vehicle is communicated to vehicle \( i \). In addition, it is assumed that there is no communication delay. Thus \( \mathcal{X}_k \sim \mathbb{D}^i \) is replaced by a time-varying set \( \mathcal{X}_k \sim E^i u_k^{i-1} \) and a new local MPC problem for every vehicle \( i \) can be formulated as

**Problem 3.2.**

\[
\begin{align*}
\min_{u_k^i} & \quad ||\bar{x}_1^i||_{P^i} + \sum_{l=0}^{N^i-1} ||\bar{x}_{l+1}^i||_{Q^i_l} + ||u_{l+1}^i||_{Q^i_u} \\
\text{s.t.} & \quad \bar{x}_{l+1}^i = A^i \bar{x}_l^i + B^i u_l^i, \quad \bar{x}_0^i = \bar{x}_1^i, \\
& \quad \bar{x}_l^i \in \mathcal{X}_k^i, \quad u_l^i \in \mathcal{U}_k^i, \quad \forall l \in \{0, ..., N^i - 1\} \\
& \quad \bar{x}_{N^i}^i \in \mathcal{T}^i, \quad \bar{x}_0^i \sim E^i u_k^{i-1}, \quad u_k^i = u_0^i,
\end{align*}
\]

where \( u_k^{i-1} \) is the input of vehicle \( i-1 \) at time \( k \). In this case, at time instant \( k \), controller \( i \) computes \( u_k^i \) after receiving \( u_k^{i-1} \) and then applies its input to the \( i \)-th vehicle; after this, \( u_k^i \) is transmitted to controller \( i+1 \) and the process is repeated. The computations of all local controllers have to be completed before the next time instant \( k+1 \), which means that sequential computation of all local controllers in one sampling period is required.

To further illustrate the difference in computation type between DeMPC Problem 3.1 and DMPC Problem 3.2, let \( t^i \) denote the computation time spent on one local optimization problem by local controller \( i \) from the DeMPC or the DMPC, for all \( i \in \{2, ..., N_A\} \). In one sampling period, the total computation time is \( t_R = \max\{t^2, ..., t^{N_A}\} \) for the DeMPC Problem 3.1 and \( t_D = \sum_{i=2}^{N_A} t^i \) for the DMPC Problem 3.2.

Then, because the constraints of new MPC Problem 3.2 become time-varying and different from Problem 3.1, recursive feasibility should be proved for Problem 3.2. This is important because if recursive feasibility holds, the constraints will be satisfied all the time if Problem 3.2 is initially feasible.
Proposition 3.4. The feasible set $X_i^F$ of Problem 3.2 is given by

$$X_i^F = Q^i[(X_i^R \sim E^i u_{i,k}^{-1}) \cap X_F(T^i, N^i - 1)] \cap X^i. \quad (3.7)$$

Proof. Based on Definition 3.1, a feasible $\bar{x}_{1|k}^i$ for Problem 3.2 satisfies:

$$\bar{x}_{1|k}^i \in (X_i^R \sim E^i u_{i,k}^{-1}) \cap X_F(T^i, N^i - 1). \quad (3.8)$$

Considering that $\bar{x}_{1|k}^i = A^i \bar{x}_{0|k}^i + B^i u_{0|k}$ and the definition of the one-step set $Q^i$ as shown in Definition 2.3, the following relationship should hold for a feasible state $x_{k}^i$:

$$x_{k}^i \in Q^i[(X_i^R \sim E^i u_{k}^{-1}) \cap X_F(T^i, N^i - 1)], \quad (3.9)$$

and, additionally, $x_{k}^i \in X^i$. Hence the feasible set is $Q^i[(X_i^R \sim E^i u_{k}^{-1}) \cap X_F(T^i, N^i - 1)] \cap X^i$. \hfill \Box

Theorem 3.5. For any $i \in \{2, \ldots, N_A\}$ and corresponding local MPC Problem 3.2, if $X_i^R$ is a robust control invariant set of subsystem $i$ within the systems (2.17) and $X_i^R \sim D^i \subseteq X_F(T^i, N^i - 1)$, then the local MPC Problem 3.2 is recursively feasible.

Proof. If Problem 3.2 is feasible at time $k$, $\bar{x}_{1|k}^i \in X_i^R \sim E^i u_{k}^{-1}$ and it can be found that

$$x_{k+1}^i = \bar{x}_{1|k}^i + E^i u_{k}^{-1} \in X_R^i.$$ 

Considering Theorem 2.6 and $X_R^i \subseteq X^i$, the following holds:

$$x_{k+1}^i \in X_R^i \subseteq \tilde{Q}^i(X_R^i) \cap X^i.$$ 

And, in addition, as shown in Remark 2.1, we can obtain

$$x_{k+1}^i \in \tilde{Q}^i(X_R^i) \cap X^i = Q^i(X_R^i \sim D^i) \cap X^i.$$ 

Since $E^i u_{k}^{-1} \in D^i$ for all $k \in \mathbb{N}$, based on the definition of Pontryagin difference, it can be found that $X_i^R \sim D^i \subseteq X_i^R \sim E^i u_{k}^{-1}$. Considering the assumption that $X_i^R \sim D^i \subseteq X_F(T^i, N^i - 1)$, we then have

$$X_i^R \sim D^i \subseteq (X_i^R \sim E^i u_{k}^{-1}) \cap X_F(T^i, N^i - 1).$$
Also, based on Remark 2.2, the following holds:

\[ Q^i(X^i_R \sim D^i) \cap X^i \subseteq Q^i([X^i_R \sim E^i u^{i-1}_k] \cap X^i_F(T^i, N^i - 1)] \cap X^i = X^i_F. \]

Therefore, \( x^i_{k+1} \in Q^i(X^i_R \sim D^i) \cap X^i \subseteq X^i_F \) and Problem 3.2 is also feasible at time \( k + 1 \). \( \square \)

Thus, the MPC Problem 3.2 is recursively feasible under the same conditions as the ones of Theorem 3.2, while using a less conservative, time-varying robustness constraint.

**Remark 3.2.** The definition of ISS does not suit Problem 3.2 anymore because the feasible region (3.7) is time-varying. Instead, stability of DMPC Problem 3.2 can be regarded as a modification of Definition 2.8, where the disturbance \( w_k \in W \) becomes \( w_k = E^i u^{i-1}_k \) which is also time-varying. This requires new definition of stability and the proof of it, which will be put as future research. In this work, we will choose parameters of Problem 3.2 based on the conditions of Theorem 3.3 except that \( X^i_R \sim D^i \) contains the origin in its interior. Then, the convergence of Problem 3.2 will be checked in the simulation. Similarly, we will skip the stability issue of the feedback DMPC Problem 4.3 in Chapter 4 as well.

By replacing \( X^i_R \sim D^i \) by \( X^i_R \sim E^i u^{i-1}_k \), the benefit is twofold: the feasible region is enlarged and \( U^{i-1} \) of Problem 3.2 is allowed to be larger than in Problem 3.1 which can lead to higher capacity of the platoon equipped with the new DMPC controllers. The first benefit is trivial and for the second one, as it can be seen in Section 3.1 for Problem 3.1, \( U^{i-1} \) is limited to achieve \( D^i \subseteq X^i_R \) such that \( X^i_R \sim D^i \) contains the origin. However, for the constraint \( X^i_R \sim E^i u^{i-1}_k \), as the distributed systems (2.13) go to the steady-state, \( E^i u^{i-1}_k \) will eventually lie in the set \( X^i_R \) and thus \( X^i_R \sim E^i u^{i-1}_k \) contains the origin even if \( E^i U^{i-1} \) is not a subset of \( X^i_R \). Therefore, the convergence of Problem 3.2 can be achieved with larger \( U^{i-1} \) compared to Problem 3.1. This shows that the DMPC algorithm has a potential to incorporate more vehicles in a platoon than the DeMPC algorithm if there is a requirement on the size of \( U^1 \) due to safety reasons.

The cost of using \( X^i_R \sim E^i u^{i-1}_k \) is that one round of communication and sequential computation are needed in every sampling period. Thus, the computation load is higher than the DeMPC algorithm and is increasing with the platoon size. In addition, the Pontryagin difference between \( X^i_R \) and \( E^i u^{i-1}_k \) needs to be calculated in real-time, which
could lead to a higher computational load. This problem can be solved by implementing the constraint $X_iR \sim E_iu_{k-1}$ using the following remark:

**Remark 3.3.** Given a polytope $\Omega \subset \mathbb{R}^n$ in hyperplane description as $\Omega = \{x|Px \leq b\}$ and a vector $\phi \in \mathbb{R}^n$, the following relationship holds:

$$\Omega \sim \phi = \{x|P(x + \phi) \leq b\}.$$

Using the above remark eliminates the need for online computation of the Pontryagin difference.

### 3.4 Distributed synthesis for plug and play

When new vehicles join a platoon or leave a platoon, two actions need to be executed to achieve plug and play operations, including the distributed synthesis of new local controllers and the initialization for the new platoon. As described in Section 2.2, our focus will lie on the distributed synthesis during plug and play operations.

There are two problems in the controller synthesis for plug and play operation, i.e. generation of the required parameters for the synthesis and the local synthesis itself. The parameter generation depends on the specific scenario, i.e. joining or leaving. When all parameters are ready, the synthesis of every new local controller follows the same procedure. In the end, we will provide two algorithms including the local algorithm for synthesis when all parameters are ready and another high level algorithm which also contains the parameter generation and the simplified initialization process.

In addition, we will only consider a simple scenario where one vehicle tries to join a platoon and the distributed synthesis is strictly local. Based on the input coupling in model (2.13), strictly local synthesis means only the controller of the vehicle which tries to join and the controller of its first follower will be redesigned, which is beneficial because only a minimal number of controllers need to be redesigned.

#### 3.4.1 Parameter generation for distributed synthesis

To synthesize either MPC Problem 3.1 or Problem 3.2 for subsystem $i$, several parameters are required, including the model $(A^i, B^i, E^i)$, the sets of local constraints $(X^i, U^i)$,
the input bound of the preceding vehicle $U_{i-1}$ which is used to construct the disturbance set $E^i U_{i-1}$. Among all the parameters, suitable input bounds shall be generated to ensure the existence of RCI sets for controller synthesis during plug and play operations.

Considering the scenario where vehicle $i - 1$ and vehicle $i + 1$ are two neighbors in the previous platoon, new subsystem $i$ sends a request to join between them later. Since we assume that the distributed synthesis is strictly local, only two controllers should be synthesized, including one for vehicle $i$ and another one for vehicle $i + 1$. Thus, the existence of the RCI sets $X^i_R$ and $X^{i+1}_R$ should be ensured for the synthesis of the two controllers.

In this scenario, $U^i$ is the only input bound which should be tuned to ensure the existence of the two RCI sets. This is due to the fact that the change of other input bounds will lead to further modifications for extra controllers. For example, the change of $U^i - 1$ and $U^{i+1}$ will lead to modification of controllers from vehicle $i - 1$ and vehicle $i + 2$, respectively. Thus, the two RCI set can be viewed as functions of $U^i$, i.e. $X^i_R(U^i)$ and $X^{i+1}_R(U^i)$.

Therefore, the goal of the parameter generation is to calculate an input bound $U^i$, where $U^i - 1 \subseteq U^i \subseteq U^{i+1}$, for the new vehicle such that $X^i_R$ and $X^{i+1}_R$ exist. Note that for MPC Problem 3.1, it is further required that $E^i U_{i-1} \subseteq X^i_R$ and $E^i U^i \subseteq X^{i+1}_R$ such that the robust constraints can contain the origin as discussed in Section 3.2.1, which is not needed for MPC Problem 3.2. If such $U^i$ cannot be found, the request for joining from the vehicle will be rejected.

The scenario where one vehicle tries to leave is even simpler than the scenario of joining because no input bound needs to be generated. Considering that vehicle $i$ is leaving and the preceding vehicle of vehicle $i + 1$ becomes vehicle $i - 1$, only the controller for vehicle $i + 1$ needs to be redesigned. In previous topology, $U^i - 1 \subseteq U^i \subseteq U^{i+1}$ holds and $X^{i+1}_R(U^i, U^{i+1})$ exists. In the new platoon, the RCI set of vehicle $i + 1$ can be denoted as $X^{i+1}_{R,new}(U^i - 1, U^{i+1})$. Since the corresponding disturbance set of $X^{i+1}_{R,new}$, i.e. $E^{i+1} U_{i-1}$, is a subset of the disturbance set of $X^{i+1}_R$, i.e. $E^{i+1} U^i$, $X^{i+1}_{R,new}$ will exist without modifying any input bound.
3.4.2 Synthesis of a local controller

Given all the required parameters, to synthesize a MPC Problem 3.1, all conditions of Theorem 3.3 shall be satisfied. As for the synthesis of MPC Problem 3.2, except that $X^i_R \sim D^i$ contains the origin in condition 1 of Theorem 3.3, other conditions of Theorem 3.3 shall still be satisfied. It can be seen that the conditions for the synthesis of the two controllers do not differ much, which motivates us to compact the synthesis of these two controllers into one algorithm. We will use a logic variable $\alpha$ to represent the controller type, i.e. $\alpha = 1$ represents MPC Problem 3.1 and $\alpha = 0$ represents MPC Problem 3.2.

Algorithm 1 is used for the controller synthesis and serves as a local algorithm of the high level algorithm for plug and play operation. The algorithm for plug and play operation should combine the results of the parameter generation, Algorithm 1 and the simplified initialization process together. For example, Algorithm 2 can be used to achieve plug and play operation in a scenario where one vehicle tries to join a platoon.

---

**Algorithm 1** Controller design for vehicle $i$

**Input:** $A^i$, $B^i$, $E^i$, $X^i$, $X^i_R$, $U^i$, $U^{i-1}$, $Q^i_x$, $Q^i_u$, $\alpha$.

**Output:** MPC Problem 3.2 or MPC Problem 3.1 for vehicle $i$ or Error message.

1: Calculate the Pontryagin difference $X^i_R \sim E^i U^{i-1}$;
2: if $\alpha == 1$
3: If $0 \notin X^i_R \sim E^i U^{i-1}$,
4: Report error ”The origin is not contained in the robustness constraint”.
5: end if
6: Calculate $K^i$, PI set $T^i$ and $P^i$ such that condition 4 of Theorem 3.3 is satisfied;
7: Increase $N^i$ iteratively until $X^i_R \sim D^i \subseteq X^i_p(T^i, N^i - 1)$ holds;
8: Formulate the MPC Problem 3.2/Problem 3.1 with all obtained parameters.
Algorithm 2 Plug and Play operation when vehicle $i$ sends request for join

**Distributed synthesis:**

Input: Join request, $U^{i-1}$, $U^{i+1}$, vehicle models, state constraints and cost matrices of vehicle $i$ and $i+1$.

Output: MPC Problems for vehicle $i$ and vehicle $i+1$ or Join request rejected.

1: Initialization $U^i = U^{i+1}$ and $0 < \delta_i < 1$ equals to a small value, e.g. 0.1;
2: while $\kappa = 1$
3: if $U^{i-1} \subseteq U^i$ then
4: if $X^i_R$ and $X^{i+1}_R$ exist then
5: Run Algorithm 1 for vehicle $i$ and $i+1$;
6: If no error message, return the two MPC problems while set $\kappa = 0$, end if.
7: else
8: Increase $\delta_i$ and let $U^i = (1 - \delta_i)U^{i+1}$.
9: end if
10: else
11: Join request rejected while setting $\kappa = 0$.
12: end if
13: end while

**Initialization:**

1: Apply merging protocols to obtain a feasible state for new platoon, e.g. the origin;
2: Apply the generated controllers to the new platoon.

3.5 Numerical analysis

In this section, the numerical analysis is conducted for both the DeMPC Problem 3.1 and the DMPC Problem 3.2. In addition, the algorithms of distributed synthesis in Section 3.4 will also be validated by simulation.

Firstly, we will introduce the parameters chosen for the two problems. Then, the simulation results of two scenarios will be discussed, including the scenario where $h^i = 0$ which is called constant spacing policy and another one where $h^i = 1$ which is called velocity dependent spacing policy.

In the first scenario, the goal is to illustrate the advantage of the DMPC algorithm without the condition that $X_R^i \sim \mathbb{D}^i$ contains the origin over the DeMPC algorithm.
which requires it. The reason why we consider this scenario separately is because this
advantage is only obvious when this condition becomes the main limitation, e.g. under
the resulting model (2.13) when \( h^i = 0 \).

In the second scenario, the goal is to compare the performance of the DeMPC Problem
3.1 and the DMPC Problem 3.2. Three attributes of performance will be analyzed,
including the size of feasible region, the computation time and the total costs which
represents the input effort and the state deviation from the origin. Additionally, the
robustness of the DMPC to the possible disturbance on the communicated information
\( u_k^{i-1} \) will also be tested.

Finally, simulation results of distributed synthesis will be presented and discussed.

3.5.1 Parameters for simulation

The sampling time \( T \) is chosen to be 1 second and the state constraints of all subsystems
are chosen to be the same as \( X_i^1 = [−d_s, 120] \times [−15, 15] \) where \( d_s = 4, \forall i \in \{2, ..., N_A\} \).
The input bound of the last vehicle is the physical limit of its desired acceleration, which
is \( U^{NA} = [−5, 3] \). For simulation, an input reference is generated for the leader within
its input bound \( U^1 \). The stage cost matrices and the input cost matrices are chosen as
\( Q^i_x = diag(1, 1) \) and \( Q^i_u = 1, \forall i \in \{2, ..., N_A\} \).

Based on Theorem 3.3, LQR control law is chosen as the terminal controller while the
terminal cost \( ||x_{N+k|k}||_P \) and the terminal constraint \( T^i \) are a Lyapunov function and
a positively invariant set for the closed-loop nominal model under the LQR controller,
respectively. In addition, the maximal RCI set, i.e. \( X_R \) of the system (2.17) may not
be finitely determined, in which case the RCI set can be replaced by an inner or outer
approximation of it [1].

3.5.2 Constant distance

The analysis in this subsection is mainly to illustrate the advantage of the DMPC al-
gorithm without the condition that \( X_R \sim D \). This advantage makes it possible for
the DMPC algorithm to achieve convergence with less conservative input bounds than
the RMPC algorithm. This is because the scaling matrix \( C \) of the DeMPC algorithm
Open-loop DeMPC and DMPC

![Figure 3.1: The evolution of $e^2_p$ in 2-vehicle platoon with $h = 0$ and $C = diag(0.9, 0.9)$.](image)

is chosen based on not only the existence of the RCI sets but also the condition that $X^i_R \sim \mathbb{D}^i$ contains the origin.

However, this advantage of DMPC is only obvious when the condition that $X^i_R \sim \mathbb{D}^i$ contains the origin is the main limit for the $C$ matrix. For example, in this work, it is obvious for the resulting model (2.14) when $h = 0$ but not clear when $h = 1$ as shown in Section 3.5.3.

When $h = 0$, we first choose the scaling matrix $C = diag(0.9, 0.9)$. For simplicity, $N_A$ equals to 2 in this scenario. The evolution of the state $e^2_p$ under the RMPC and the DMPC is shown in Figure 3.1. The horizontally red line is $-d_s$ which is the chosen minimal safety distance. It can be found that with $C = diag(0.9, 0.9)$, as reported in [14], convergence of the state to the origin can not be achieved under the RMPC algorithm. This is because that the values of the entries in $C$ are too large and thus the robustness constraint $X^i_R \sim \mathbb{D}^i$ does not contain the origin. By contrast, the DMPC algorithm allows us to choose the entries of $C$ with large values, which leads to a less conservative $U^1$ for the leading vehicle.

The convergence issue of the RMPC algorithm can be solved by using entries with smaller values in $C$ matrix. In Figure 3.2, three examples of $X^2_R \sim \mathbb{D}^2$ with difference $C$ matrices can be found. It can be seen that $X^2_R \sim \mathbb{D}^2$ contains the origin when $C = diag(0.6, 0.9)$ and $C = diag(0.4, 0.4)$. The cost is that the allowable range of the inputs for the leading vehicle is smaller. For the RMPC algorithm, to enlarge the robustness set, the entries of $C$ shall have the largest values while satisfying $\mathbb{D}^2 \subseteq X^2_R$. 
3.5.3 Velocity-dependent distance

In this part, the time gap $h$ is chosen to be 1 second, which implies the desired distance between two vehicles also depends on the velocity as shown in equation (2.12). The input scaling matrix $C$ is chosen to be $\text{diag}(0.9, 0.9)$ for both the DeMPC algorithm and the DMPC algorithm. 5 vehicles in a platoon are considered for simulation. The simulation results are shown in Figure 3.3. In the first column, $p^1$ denotes the absolute position of the leading vehicle and other plots show the change of the state $e_p^i$. The evolution of the velocities of every vehicle is shown in the second column while the input profiles are shown in the last column. It can be seen that the states of all vehicle are steered to the origin by both algorithms. Compared to the results when $h = 0$, the convergence of the DeMPC can be also achieved with $C = \text{diag}(0.9, 0.9)$ due to the difference in $B$ matrix caused by the different value of $h$.

Three attributes of the performance will be compared, including the feasible region, the computation time and the total cost.

Most importantly, the feasible region of the DMPC algorithm is larger than the one of the DeMPC algorithm. As an example, the feasible sets of the 5th vehicle under the DMPC algorithm and the RMPC algorithm are shown in Figure 3.4. It can be found that the feasible region for the RMPC is a subset of the one of the DMPC algorithm. Noticing that the feasible region of the DMPC is time-varying as shown in equation (3.7), the set of the DMPC in Figure 3.4 is chosen at one time instant. By contrast, the feasible set for the RMPC is time invariant.
Figure 3.3: Simulation results for 5 vehicles when $h = 1$.

Figure 3.4: Feasible region for the 5th vehicle.
Secondly, the total costs of the two control schemes are compared. The cost is defined as follows:

\[
\sum_{i=2}^{N_A} \frac{t/T}{T} \sum_{k=1}^T \left[ \tilde{x}_i^T Q_1 \tilde{x}_i^k + \tilde{u}_i^T Q_2 \tilde{u}_i^k \right],
\]

(3.10)

where \( t \) is the total simulation time, \( \tilde{x}^i \) and \( \tilde{u}^i \) represent the normalized states and inputs by the full range of their corresponding constraint. The defined cost penalizes the state deviation from the origin, which is related to the safety and the convergence speed, and the input effort which implies the fuel consumption. Since we assume the two types of cost are of equal importance, both \( Q_1 \) and \( Q_2 \) are chosen to be identity matrices with suitable dimension. It is found that the total cost of the DMPC is 16.5% higher than the cost of the decentralized control using the RMPC. Therefore, the DMPC algorithm leads to a poorer performance than the DeMPC algorithm.

For the third attribute of the performance, the computation time of the algorithms is compared when the platoon contains 5 vehicles and all the MPC algorithms are implemented in YALMIP toolbox. The computation time spent in one optimization problem of a local controller from the DeMPC and the DMPC algorithm is 9.2 and 8.7 milliseconds on average, respectively. The total computation in one sampling period is \( t_R = 11.5 \) ms for the DeMPC and \( t_D = 43.5 \) ms for the DMPC. Since sampling time \( T \) equals to 1 second, both \( t_R \) and \( t_D \) are smaller than \( T \). However, the DMPC algorithm is slower than the DeMPC due to the fact that sequential computation has to be conducted for all local controllers of the DMPC in one sampling period. In addition, note that the computation time of the DMPC will increase with the increase in platoon size.

Additionally, to test the robustness of the DMPC algorithm against the error in the communicated information, the simulation is conducted using \( u^{i-1} + \delta^i \) as the information received by vehicle \( i \) instead of \( u^{i-1} \), where \( \delta^i \in [-1, 1], \forall i \in \{2, \ldots, N_A\} \), is a random error. Note that recursive feasibility with this perturbed communication also depends on the initial conditions of the vehicles; for a challenging set of initial conditions, the simulation was conducted for 5 vehicles with and without communication errors. As an example, the change of the state \( e_p^{\delta} \) and the input profile \( u^{\delta} \) are shown in Figure 3.5. It can be seen that the control problem with communication error is still feasible. However, it is worth mentioning that it is possible for the problem to become unfeasible with certain initial conditions. By tuning \( C \), the change of the robustness constraint and thus the feasible region can also influence the robustness.
Remark 3.4. To achieve parallel computation for the DMPC algorithm, one solution is to use the robustness constraint \( X^i_R \sim E^i u^{i-1}_{k-1} \) in Problem 3.2 at time instant \( k \), which uses the \( u^{i-1} \) input from the previous time instant \( k-1 \). In this case, recursive feasibility can not be guaranteed, but there is a good chance for the constraint to be feasible by continuity of linear MPC laws. Most importantly, the total computation time in one sampling period is reduced since all problems can be solved in parallel as in the DeMPC algorithm.

Based on the analysis above, the DMPC algorithm has poorer performance but yields larger feasible regions than the DeMPC algorithm. Under the constant safe distance model when \( h = 0 \), the DMPC algorithm allows to have a scaling matrix with larger entries and thus leads to larger input ranges for the vehicles, which implies that the DMPC has a potential to incorporate more vehicles in a platoon at the cost of the increase in the computation time.

### 3.5.4 Distributed synthesis

To validate Algorithm 1 and 2 in Section 3.4, the scenario where a vehicle tries to join a three-vehicle platoon between the second vehicle and the third vehicle is simulated. Only the DeMPC algorithm is considered due to its similarity to the DMPC algorithm. The simulation results are shown in Figure 3.6.

The vehicle which tries to join is denoted as vehicle \( j \). Other vehicles are named based on the sequence in previous platoon. After the finish of joining, vehicle 3 in the previous platoon becomes the fourth vehicle and vehicle \( j \) is the third vehicle in the new platoon.
The join request is sent at time $t = 50$ which is denoted as a green vertical line in Figure 3.6.

Before that time, as shown in the figure, the state $x_j$ of vehicle $j$ is not defined because it has not join the platoon. In addition, $e_p^3$ still contains the relative information between vehicle 3 and vehicle 2 in the previous topology.

After the join request is sent, Algorithm 2 starts to run, which synthesizes two new controllers for vehicle $j$ and vehicle 3. Then, an initialization process is triggered to set new feasible states for vehicle $j$ and vehicle 3. For simplification, the initialization process is just directly setting of the values of new states. Finally, the obtained controllers are applied to the vehicle $j$ and vehicle 3. Note that $e_j^j$ only starts to appear when $t = 50$ because vehicle $j$ is one subsystem of the platoon after that time. In addition, $e_p^3$ contains the relative information between vehicle 3 and vehicle $j$ in the new platoon.

Figure 3.6: Simulation of a vehicle joining a platoon when time=50

### 3.6 Conclusion

In this chapter, the decentralized control using RMPC algorithm in [1] was applied to the decentralized control problem of vehicle platooning, which led to a DeMPC scheme. The proof of ISS which was not included in [1] was provided for the RMPC algorithm.
Then, a DMPC algorithm was developed with a new time-varying robustness constraint. The proof of recursive feasibility was provided. In terms of its performance compared to the DeMPC Problem 3.1, its feasible region was larger and its robustness against communication errors was observed. The price paid for these improvements is, however, that sequential computation are required for the DMPC algorithm which leads to higher computational load than the DeMPC and the total cost are increased.

Based on the above conclusion, several properties can be expected from an ideal DMPC algorithm, including the parallel computation and the communication of a bound around the input $u_{i-1}^k$ instead of the exact value of $u_{i-1}^k$. The parallel computation is preferred for DMPC algorithms because the computation time will be independent on the platoon size. As for the communication, due to the possible disturbance, it is more robust to communicate a bound around the input $u_{i-1}^k$ instead of the exact value.

These properties cannot be achieved by distribution of the RMPC algorithms which use nominal model in the constraints. In these algorithms, the bounds which should be satisfied by the real state are normally modified for the nominal state, such as $X_{R}^i \sim D_{i}$ in MPC Problem 3.1 and the sets in [7]. Therefore, set operations such as Pongryagin difference and Minkowski sum are required. If a bound around the input $u_{i-1}^k$ is communicated among the robust controllers in this type, either online or pre-computed set operations between one set and one time-varying set will be needed, e.g. the effort made by the authors of [14] and the preliminary research in Appendix A to adapt the RCI sets online. These methods will lead to huge computational effort or even infeasibility as shown in [14]. Thus, to avoid online set operation when the bound is communicated, the perturbed model (2.17) should be directly used in the constraints, which motivates the research on the robust algorithm in [2].
Chapter 4

Feedback DeMPC and DMPC

In this chapter, the robust MPC algorithm from [2] will be studied. It uses the perturbed model in the constraints and uses the nominal model in cost function which avoids the min-max optimization. In addition, instead of the open-loop input sequence prediction, feedback policies are predicted in the algorithm, which is less conservative than open-loop input sequence prediction used in MPC Problem 3.1.

The algorithm will be first applied to the decentralized control problem and then modified to obtain a novel DMPC algorithm by incorporating communication. The distributed synthesis and the numerical analysis will also be addressed. In the end, the algorithms in this chapter will be compared with the algorithms in Chapter 3.

4.1 Decentralized control with feedback RMPC

4.1.1 Input parameterization

Define the input, state and disturbance vectors $u_k^i \in \mathbb{R}^{mN_i}$, $x_k^i \in \mathbb{R}^{n(N_i+1)}$ and $w_k^i \in \mathbb{R}^{nN_i}$, respectively, as

\[
\begin{align*}
    u_k^i & \triangleq [u_{i0k}, \ldots, u_{i(N_i-1)k}]^T, \\
    x_k^i & \triangleq [x_{i0k}, \ldots, x_{i(N_i)k}]^T, \\
    w_k^i & \triangleq [w_{i0k}, \ldots, w_{i(N_i-1)k}]^T.
\end{align*}
\]
For subsystem $i$, the predicted input $u_{l|k}^i$ is parameterized as a affine function of the past disturbances as follows

\begin{align}
    u_{0|k}^i &= v_{0|k}^i, \\
    u_{l|k}^i &= h_{l|k}^i(w_{0|k}^i, ..., w_{l-1|k}^i) = \sum_{j=0}^{l-1} M_{(l,j)|k}^i w_{j|k}^i + v_{l|k}^i, \quad \forall l \in \{1, ..., N^i - 1\},
\end{align}

where $N^i$ denotes the prediction horizon. To lump the two equations together, the first input $u_{0|k}^i$ is denoted as $\sum_{j=-1}^{0} M_{(0,j)|k}^i w_{j|k}^i + v_{0|k}^i$, where $M_{(0,0)|k}^i$ and $M_{(0,-1)|k}^i$ are matrices with all entries equal to zero. In addition, the terminal control law is represented by $h_{t}^i(x_{N^i|k}^i)$. A matrix $M_{k}^i \in \mathbb{R}^{mN^i \times nN^i}$ and a vector $v_{k}^i \in \mathbb{R}^{mN^i}$ can be defined:

\begin{align}
    M_{k}^i &\triangleq \begin{bmatrix}
        0 & \cdots & \cdots & 0 \\
        M_{(1,0)|k}^i & 0 & \cdots & 0 \\
        \vdots & \ddots & \ddots & \vdots \\
        M_{(N^i-1,0)|k}^i & \cdots & M_{(N^i-1,N^i-2)|k}^i & 0
    \end{bmatrix}, \quad v_{k}^i \triangleq \begin{bmatrix}
        v_{0|k}^i \\
        \vdots \\
        \vdots \\
        v_{N^i-1|k}^i
    \end{bmatrix}.
\end{align}

4.1.2 MPC formulation

The robust MPC algorithm with the parameterized input can be directly applied to every subsystem represented by equation (2.17). The obtained DeMPC formulation is as follows

**Problem 4.1.**

\[
\min_{M_{k}^i, u_{k}^i} \|\tilde{x}_{N^i|k}^i\|_P + \sum_{l=0}^{N^i-1} \|\tilde{x}_{l|k}^i\|_{Q^i} + \|u_{l|k}^i\|_{Q^u} \\
\text{s.t.} \quad x_{l+1|k}^i = A^i x_{l|k}^i + B^i u_{l|k}^i + w_{l|k}^i, \quad x_0^i = x_k^i \\
    u_{l|k}^i = \sum_{j=0}^{l-1} M_{(l,j)|k}^i w_{j|k}^i + v_{l|k}^i, \quad u_0^i = u_{0|k}^i, \\
    x_{l|k}^i \in X^i, \quad u_{l|k}^i \in U^i, \quad \forall w_{l|k}^i \in D^i, \quad \forall l \in \{0, ..., N^i - 1\} \\
    x_{N^i|k}^i \in T^i.
\]
where $T_i$ is a set to constrain the predicted terminal state. Note that the nominal model (2.18) is used for the cost function and the perturbed model (2.17) is used in the constraints.

To guarantee recursive feasibility and ISS, $T_i$ is chosen to be the maximum RPI set of the closed loop system $x_{i,k+1}^i = (A^i + B^iK^i)x_{i,k}^i + w_{i,k}^i$, where $K^i$ is chosen such that $A^i + B^iK^i$ is Schur. Therefore, to guarantee the existence of the RPI set, the scaling of the input bounds $U_i$ has to be conducted to ensure that the disturbance set is small enough as shown in Section 3.1. Note the terminal set here is a RPI set for the perturbed model, by contrast, the terminal sets in Problem 3.2 and Problem 3.1 are PI sets for the nominal model. Based on the results in [2], ISS is guaranteed by choosing terminal cost matrix $P_i$ so that $||\bar{x}_{N_i,k}||_P$ is a Lyapunov function in the terminal set $T_i$ for the nominal closed-loop system $\bar{x}_{k+1}^i = (A^i + B^iK^i)\bar{x}_k^i$.

In fact, based on Theorem 23 in [2], if condition 2 to 4 of Theorem 3.3 are satisfied while modifying the $T_i$ to be a RPI set for the perturbed closed-loop system in condition 3, recursive feasibility and ISS of MPC Problem 4.1 can be guaranteed.

4.2 Distributed control with feedback DMPC

The conservativeness in Problem 4.1 is introduced by the robustness against all possible values of the disturbances inside the set $D_i$. It can be expected that the conservativeness could be reduced if we can reduce the size of disturbance sets by establishing communication. Therefore, it is reasonable to make the disturbance sets time varying such that the new sets are subsets of $D_i$ while recursive feasibility can still be satisfied under the time-varying constraints.

In this section, we will first analyze the conditions on time-varying disturbance sets for recursive feasibility. Then, a new MPC formulation will be promoted with updating of disturbance sets while recursive feasibility is still satisfied.

4.2.1 Recursive feasibility under time-varying disturbance sets

To make the approach more clear, assume that the prediction horizons of all local controllers are identical and a sequence of disturbance bounds at time $k$ over the prediction
horizon $N$ is defined as
\[
\mathcal{W}_k^i = \{\mathcal{D}_{0,k}^i, ..., \mathcal{D}_{N-1,k}^i\},
\]
(4.4)
where $\mathcal{D}_{l,k}^i \subseteq \mathcal{D}^i$ is a set to constrain the disturbance $w_{l|k}^i$, $\forall l \in \{0, ..., N-1\}$. The MPC problem can be reformulated as follows

**Problem 4.2.**

\[
\begin{align*}
\min_{M_i^k, u_i^k} & \quad ||\hat{x}_{N|k}^i||_{P^i} + \sum_{l=0}^{N-1} ||\hat{x}_{l|k}^i||_{Q^i} + ||u_{l|k}^i||_{Q^i} \\
\text{s.t.} & \quad x_{l+1|k}^i = A^i x_{l|k}^i + B^i u_{l|k}^i + w_{l|k}^i, \quad x_{0|k}^i = x_{k}^i, \\
& \quad u_{l|k}^i = \sum_{j=0}^{l-1} M_{(l,j)|k}^i w_{j|k}^i + v_{l|k}^i, \quad u_{0|k}^i = u_{0|k}^i, \\
& \quad x_{l|k}^i \in \mathcal{X}^i, \quad u_{l|k}^i \in \mathcal{U}^i, \quad \forall w_{l|k}^i \in \mathcal{D}_{l|k}^i, \quad \forall l \in \{0, ..., N-1\} \\
& \quad x_{N|k}^i \in \mathcal{T}^i.
\end{align*}
\]

It can be seen that if $\exists l \in \{0, ..., N-1\}$ such that $\mathcal{D}_{l|k}^i \subseteq \mathcal{D}^i$, Problem 4.2 will have a larger feasible region than Problem 4.1. This is because the disturbance which should be compensated belongs to a set in smaller size. For Problem 4.2, the following theorem holds:

**Theorem 4.1.** If Problem 4.2 is feasible at time $k$ and the following conditions are satisfied:

\[
\begin{align*}
& w_{k}^i \in \mathcal{D}_{0|k}^i, \quad (4.5a) \\
& \mathcal{D}_{l-1|k+1}^i \subseteq \mathcal{D}_{l|k}^i, \forall l \in \{1, ..., N-1\}, \quad (4.5b)
\end{align*}
\]

then Problem 4.2 is also feasible at time $k + 1$. 

Proof. If Problem 4.2 is feasible at time $k$, the solution is a sequence of feedback policies \( \{v^i_{0|k}, h^i_{1|k}, \ldots, h^i_{N-1|k}\} \) as shown in equation (4.2) and the following holds

\[
v^i_{0|k} \in U^i;
\]

\[
x^i_{1|k} \in X^i_{1|k} = \{x|x = A^i x^i_k + B^i v^i_{0|k} + w^i_{0|k}, \forall w^i_{0|k} \in D^i_{0|k}\} \subseteq X^i,
\]

\[
u^i_{1|k} \in U^i_{1|k} = \{u|u = h^i_{1|k}(w^i_{0|k}), \forall w^i_{0|k} \in D^i_{0|k}\} \subseteq U^i;
\]

\[
x^i_{2|k} \in X^i_{2|k} = A^i x^i_{1|k} + B^i u^i_{1|k} \in X^i,
\]

\[
u^i_{2|k} \in U^i_{2|k} = \{u|u = h^i_{2|k}(w^i_{0|k}, u^i_{1|k}), \forall (w^i_{0|k}, u^i_{1|k}) \in D^i_{0|k} \times D^i_{1|k}\} \subseteq U^i;
\]

\[
\vdots
\]

\[
u^i_{N-1|k} \in U^i_{N-1|k} = \{u|u = h^i_{N-1|k}(w^i_{0|k}, \ldots, w^i_{N-2|k}), \forall (w^i_{0|k}, \ldots, w^i_{N-2|k}) \in D^i_{0|k} \times \ldots \times D^i_{N-2|k}\} \subseteq U^i;
\]

\[
x^i_{N|k} \in X^i_{N|k} = A^i x^i_{N-1|k} + B^i u^i_{N-1|k} \subseteq T^i.
\]

At time $k+1$, $w^i_k$ can be obtained as $w^i_k = x^i_{k+1} - (A^i x^i_k + B^i v^i_{0|k})$ and the shifted sequence of the control policies is \( \{h^i_{1|k}; w^i_k, \ldots, h^i_{N-1|k}; w^i_k, h^i_t\} \) where

\[
h^i_{1|k; w^i_k} = M^i_{1,0} w^i_k + v^i_{0|k} = v^i_{0|k+1}, \quad (4.7a)
\]

\[
h^i_{2|k; w^i_k} = M^i_{2,0} w^i_k + M^i_{2,1} v^i_{0|k} + v^i_{2|k} = M^i_{1,0} w^i_k + v^i_{1|k+1}, \quad (4.7b)
\]

\[
\vdots
\]

\[
h^i_{N-1|k; w^i_k} = \sum_{j=1}^{N-2} M^i_{(N-1,j)} w^j_{j-1|k+1} + (v^i_{N-1|k} + M^i_{(N-1,0)} w^i_k) \quad (4.7d)
\]

\[
= \sum_{j=0}^{N-3} M^i_{(N-2,j)} w^j_{j+1|k+1} + v^i_{N-2|k+1}, \quad (4.7e)
\]

\[
h^i_t = K^i x^i_{N-1|k+1}. \quad (4.7f)
\]
Based on the assumption that the $w_k^i \in \mathbb{D}^i_{0|k}$ and $\mathbb{D}^i_{l-1|k+1} \subseteq \mathbb{D}^i_l$, $\forall l \in \{1, ..., N - 1\}$, feasibility of the shifted sequence at time $k + 1$ can be analyzed as follows:

$$x_{k+1}^i = A^i x_k^i + B^i u_{0|k}^i + w_k^i \in \mathbb{X}_{1|k} \subseteq \mathbb{X}^i,$$

$$u_{0|k+1}^i = h_{1|k;w_k^i}^i \in \mathbb{U}_{1|k} \subseteq \mathbb{U}^i;$$

$$x_{l|k+1}^i \in \{ x | x = A^i x_{l|k+1}^i + B^i u_{0|k+1}^i + w_{0|k+1}^i \in \mathbb{D}^i_{0|k+1} \subseteq \mathbb{D}^i_{l|k} \} \subseteq \mathbb{X}_{2|k} \subseteq \mathbb{X}^i,$$

$$u_{l|k+1}^i = h_{2|k;w_{0|k+1}^i}^i \in \mathbb{U}_{2|k} \subseteq \mathbb{U}^i;$$

$$\vdots$$

$$u_{N-2|k+1}^i \in \mathbb{U}_{N-1|k};$$

$$x_{N-1|k+1}^i \in \mathbb{T}^i;$$

$$x_{N|k+1}^i = A^i x_{N-1|k+1}^i + B^i K^i x_{N-1|k+1}^i + w_{N-1|k+1}^i \in \mathbb{T}^i.$$

In conclusion, because all constraints can be satisfied when applying the shifted sequence of control policies, Problem 4.2 is also feasible at time $k + 1$.

To enlarge the feasible region while guaranteeing recursive feasibility, updating of the disturbance sets of Problem 4.2, i.e. the updating of $\{\mathbb{D}^i_{0|k}, ..., \mathbb{D}^i_{N-1|k}\}$, shall be designed such that the conditions of Theorem 4.1 are satisfied and $\exists l \in \{0, ..., N - 1\}$ such that $\mathbb{D}^i_{l|k} \subseteq \mathbb{D}^i$.

### 4.2.2 Distributed MPC with updating of disturbance sets

The proposed MPC formulation with the updating of disturbance sets are shown as follows

**Problem 4.3.**

$$\min_{M^i_{j,k}, v^i_{j,k}} \| \tilde{x}_{N|k}^i \|_{P^i} + \sum_{l=0}^{N-1} \| \tilde{x}_l^i \|_{Q^i_l} + \| u_{l|k}^i \|_{Q^i_k},$$

s.t.

$$x_{l+1|k}^i = A^i x_l^i + B^i u_{l|k}^i + w_l^i, \quad x_0^i = x^i_k,$$

$$u_{l|k}^i = \sum_{j=0}^{l-1} M_{(j,l)|k}^i w_{j|k}^i + v_{l|k}^i, \quad u_{l|k}^i = v_{0|k}^i,$$

$$x_{l|k}^i \in \mathbb{X}^i, \quad u_{l|k}^i \in \mathbb{U}^i, \quad \forall w_{l|k}^i \in \mathbb{D}^i_{l|k}, \quad \forall l \in \{0, ..., N - 1\},$$

$$v_{0|k} - h_{1|k-1}^i w_{k-1}^i \in \mathbb{U}_{r}, \quad x_{N|k}^i \in \mathbb{T}^i,$$
where for all $k \geq 1$ and all $i \in \{2, \ldots, N_A\}$, $w_{k-1}^i$ can be obtained as $w_{k-1}^i = x_k^i - (A^i x_{k-1}^i + B^i w_{k-1}^i)$ and $h_{1|k-1;w_{k-1}^i}^i$ is calculated in the same way as shown in equation (4.7a), i.e. $h_{1|k-1;w_{k-1}^i}^i = M_{0(0)|k-1}^i w_{k-1}^i + v_{1|k-1}^i$. In addition, $U_l^i$ is a set to be designed and it should be convex, compact and also contains the origin as an interior-point.

As for the updating of the disturbance sets, at each time instant, only the first disturbance set $D_{0|k}^i$ in the sequence will be updated. For subsystem $i$ at time $k$, Problem 4.3 will use a sequence of disturbance sets as follows

**Proposition 4.2.** For all $k \in \mathbb{N}$ and $k \geq 1$, if Problem 4.3 is feasible at time $k - 1$ and the disturbance sets are updated as follows:

\[
D_{0|k}^i = E^i [(U_r^{i-1} \oplus h_{1|k-1;w_{k-1}^i}^i) \cap U_r^{i-1}], \quad (4.9a) \\
D_{l|k}^i = E^i U_r^{i-1}, \quad \forall l \in \{1, \ldots, N - 1\}, \quad (4.9b)
\]

then Problem 4.3 is also feasible at time $k$.

**Proof.** In this case, due to the fact that $w_k^i = E^i v_{0|k}^{i-1}$, $v_{0|k}^{i-1} - h_{1|k-1;w_{k-1}^i}^{i-1} \in U_r^{i-1}$ and $v_{0|k}^{i-1} \in U_r^{i-1}$, the condition that $w_k^i \in D_{0|k}^i$ of Theorem 4.1 is satisfied. In addition, $D_{0|k}^i = E^i [(U_r^{i-1} \oplus h_{1|k-1;w_{k-1}^i}^i) \cap U_r^{i-1}] \subseteq E^i U_r^{i-1} = D_{1|k-1}^i$ and $D_{l|k-1}^i = D_{l|k-1}^i = E^i U_r^{i-1}, \forall l \in \{2, \ldots, N - 1\}$, hence, the condition of Theorem 4.1 that $D_{l|k-1}^i \subseteq D_{l|k-1}^i, \forall l \in \{1, \ldots, N - 1\}$ is also satisfied.

Furthermore, considering the shifted sequence $\{h_{1|k-1;w_{k-1}^i}^i, \ldots, h_{N-1|k-1;w_{k-1}^i}^i, h_t^i\}$, if Problem 4.3 is feasible at time $k - 1$ and then the first element of shifted input sequence equals to $h_{1|k-1;w_{k-1}^i}^i$ similarly as shown in equation (4.7a), which indicates that the new constraint $v_{0|k}^{i-1} - h_{1|k-1;w_{k-1}^i}^i \in U_r^i$ is feasible at time $k$ if $U_r^i$ contains the origin. In conclusion, since the hypothesis of Theorem 4.1 is satisfied and since one can always set $v_{0|k}^{i-1} = h_{1|k-1;w_{k-1}^i}^i$ is feasible at time $k$, Problem 4.3 is feasible at time $k$ using the updated disturbance sets.

Therefore, recursive feasibility still holds for Problem 4.3 with the updating of disturbance sets shown in equation (4.9). Furthermore, parallel computation, i.e. all local controllers compute local problems in parallel during one sampling period, can be achieved because the updating of disturbance set $D_{0|k}^i$ requires only one-step delay information $w_{k-1}^{i-1}$, which is an advantage over the DMPC algorithm from Chapter 3.
Remark 4.1. It is not always true that the feasible region of Problem 4.3 is larger than the feasible region of Problem 4.1 because of the extra constraint $v^i_{0|k} - h^i_{1|k-1;w_{k-1}} \in U^i_r$. Even though the size of disturbance set $D^i_{0|k}$ is reduced by communication, the new constraint may introduce extra conservativeness which compensates the benefit brought by the communication. Thus, the set $U^i_r$ should be carefully tuned such that feasible region becomes larger.

4.3 Distributed synthesis for Plug and Play

For the distributed synthesis of MPC Problem 4.1 and Problem 4.3, the scenario in Section 3.4 will be used. Actually, only a minor modification is needed for the discussion in Section 3.4 to suit the feedback DeMPC and DMPC.

For the parameter generation, the main difference is that instead of ensuring the existence of RCI sets by searching for $U^i$, the existence of the RPI sets $T^i$ and $T^{i+1}$ shall be guaranteed. In addition, the set $U^i_r$ of the new vehicle can be the same as the one used for other vehicles.

For the algorithm of controller synthesis, only condition 2 to 4 of Theorem 3.3 will be needed. However, note that in condition 3, $T^i$ shall be modified to be a RPI set for the perturbed system under a stabilizing control law.

Due to these minor differences, the algorithms and their validation will not be provided in this work and the framework in Section 3.4 can still be used.

4.4 Numerical analysis

In this section, we will first introduce the parameters chosen for the simulation and implementation of the feedback MPC algorithms from this chapter. This is followed by the analysis of simulation results, where the performance of the feedback DeMPC and feedback DMPC will be compared in terms of size of the feasible region, the computation time and the total costs of input effort and state deviation from the origin.
4.4.1 Parameters and implementation

For the parameters, the state constraint, the physical limit of input $U_{NA}$, the sampling time and the cost matrices are chosen to be the same as the values in Section 3.5. The time gap $h$ is 1 second and the input scaling matrix $C$ is $\text{diag}(0.9, 0.9)$. The prediction horizons of all local controllers are $N = 3$. Let $N_A = 3$ and $U_i^r$ is tuned to be $[-1, 3]$.

As for the implementation of Problem 4.1, it can be reformulated into a QP as show in [2]. Each follower will have a local Problem 4.1 and every local problem calculates the input in one sampling time in parallel. For the implementation of Problem 4.3, it can be formulated into a QP but also has time-varying blocks in the resulting block matrices of the equality and inequality constraints. Thus, the time invariant blocks should be stored, which will be compacted with the time-varying parts at each time instant, and then a time-varying QP can be obtained online.

In addition, for Problem 4.3, it is assumed that the leader only has an input bound $U^1$ and cannot be equipped with the time varying input constraint $v_{0|k}^1 - h_i^1_{0|k-1; w_{k-1}^i} \in U^1_i$. Thus, when using the feedback DMPC algorithm for the platooning control, the first follower cannot update its disturbance bound and has to use Problem 4.3 with disturbance sets in full size, i.e. $D_{0|k}^i = E_i U_i^{-1}$ for all $l \in \{0, ..., N - 1\}$. Note that in this case, the proof of recursive feasibility in Proposition 4.2 still holds. Then, the other followers can make use of the MPC Problem 4.3 with the updating of the disturbance bounds. The analysis of the platoon using DMPC will focus on the vehicles which are equipped with Problem 4.3.

4.4.2 Simulation results

The simulation results are shown in Figure 4.1. It can be seen that both decentralized control using the MPC Problem 4.1 and the distributed control using the MPC Problem 4.3 can achieve the control goal. In fact, the results of the two different methods have similar trajectories in the figure.

To compare the performance, the cost is firstly calculated as shown in equation (3.10). The resulting cost of DMPC is 0.2% higher than it of the DeMPC, which shows the comparable performance between two methods.
The computation of a local optimization problem from the DMPC takes 73.5 milliseconds on average compared to 15.2 for the DeMPC. This large increase in computational effort is due to the online generation of time-varying constraints which need to be reformulated into parameters of the time-varying QP. However, note that the computation time of the feedback DMPC can be reduced by improving the implementation.

The advantage of the DMPC over the DeMPC lies on the reduced conservativeness, i.e. larger feasible region due to the disturbance sets with reduced size. Since the computation of the feasible region for the feedback MPC is not tractable due to the high dimension of the resulting optimization problems, the advantage will be demonstrated by the existence of feasible initial states which are feasible for the DMPC but infeasible for the DeMPC. For example, given the initial states for the first two vehicles, it is found that the state $[7.5, 0]^T$ is feasible for the third vehicle with the DMPC but infeasible for the third vehicle with the DeMPC. For the DeMPC, it is feasible when $x_3^3 = [7, 0]^T$ and the feasibility will break when $x_3^3 = [7.1, 0]^T$. In this specific case, the maximum feasible relative distance is enlarged by 0.5 meter for the third vehicle when the relative velocity is zero.
In another example, when there are 4 vehicles in the platoon and the initial conditions are the same for the first two vehicles, it is found that given zero initial relative speed for the last two vehicles, the upper bounds on the feasible relative distance are 6.7 for the third vehicle and 7.5 for the forth vehicle in the platoon using the DMPC algorithm. The bounds are 6.3 and 7 respectively using the DeMPC algorithm.

However, it is worth mentioning that even if the feasible range of the relative distance is enlarged for the last two vehicles equipped with the DMPC algorithms, the feasible region of the second vehicle is reduced because the corresponding controller uses the disturbance sets in full size and also has extra time-varying constraint compared to the RMPC. The scenario where the distributed control does not introduce conservativeness to the second vehicle is when the first vehicle also has a time-varying constraint on the input. Thus, the second vehicle can also update its disturbance set in smaller size. This scenario is possible when there is a limitation on the change of the acceleration due to the requirement of comfort or fuel consumption.

4.5 Comparison between feedback and open-loop MPC

In this section, the feedback MPC in this chapter and the open-loop MPC in Chapter 3 will be compared in terms of computation time, cost and feasible region. Note that mainly the DeMPC algorithms, i.e. Problem 3.1 and Problem 4.1, will be compared. Due to the similarity of the DMPC to the corresponding DeMPC, the difference between the two DeMPC algorithms can be naturally extended to the two DMPC algorithms.

To compare Problem 3.1 and Problem 4.1, two scenarios will be discussed separately. The first one is when the prediction horizons are the same for the two MPC algorithms, which leads to clear difference in computation time between the two algorithms. Another scenario is when the prediction horizons are chosen differently for the two problems, which results into comparable computational effort. In addition, it is assumed that there are 3 vehicles in the platoon.

As for the difference in the two DMPC algorithms, only the different computation types, i.e. sequential computation for Problem 3.2 and parallel computation for Problem 4.3, will be discussed in the end of this section.
4.5.1 Under the same prediction horizon

Firstly, it is assumed that the prediction horizons for the open-loop DeMPC Problem 3.1 and the feedback DeMPC Problem 4.1 are the same. In this case, the optimization problem of the open-loop RMPC is smaller in size than the one of the feedback RMPC. For example, when \( N^i = 11 \), the resulting QP has 11 optimization variables for the open-loop RMPC compared to 3289 variables of the feedback RMPC. The consequence is the clear difference in the computation time required to solve one local optimization, i.e. 9.2 and 468 milliseconds on average, respectively.

As for the cost, given the same initial conditions and cost matrices, the cost of the decentralized control using open-loop RMPC and the feedback RMPC are 6.6397 and 6.6396 respectively, which shows comparable performance of the two algorithms.

For the size of feasible region, it is found that the feasibility of the open-loop RMPC will hold when \( x_0^3 = [120 \ 3.1]^T \). By contrast, the feedback RMPC is feasible when \( x_0^3 = [15 \ 3.1]^T \) but infeasible when \( x_0^3 = [16 \ 3.1]^T \). This indicates that the vehicles using the open-loop DeMPC are more flexible to choose their initial conditions, especially the initial relative distance.

4.5.2 Under different prediction horizons

To achieve a comparable computation time for a local optimization problem, the prediction horizon of the feedback DeMPC should be reduced. When \( N^i = 3 \), the computation time of one local optimization in the feedback DeMPC is 15.2 milliseconds on average with 297 optimization variables. It is still around twice as much as the computation time of the open-loop RMPC with \( N^i = 11 \).

Remark 4.2. For the open-loop algorithm, the prediction horizon has a lower bound \( N_{\text{min}}^i \) in order to satisfy the condition that \( X_R^i \sim D_i \subseteq X_F(T^i, N_{\text{min}}^i - 1) \). Under the parameters chosen, \( N^3 \) has to be bigger than 11. In contrast, \( N^3 \) can be smaller, e.g. \( N^3 = 3 \), in the feedback algorithms.

With the similar computation effort, i.e. when \( N^i = 3 \) for the feedback DeMPC and \( N^i = 11 \) for the open-loop DeMPC, it can be expected that the feasible region of the feedback algorithm can be much smaller than it of the open-loop algorithm.
addition, the total cost of the feedback DeMPC does not increase clearly with the reduced prediction horizon and is still comparable to it of the open-loop DeMPC.

In addition, when $N^i = 3$ for the feedback DMPC and $N^i = 11$ for the open-loop DMPC, the total cost is 6.3112 for the feedback algorithm and 6.3086 for the open-loop DMPC, which shows comparable total costs.

### 4.5.3 Computation types of two DMPC algorithms

The main difference between open-loop DMPC Problem 3.2 and the feedback DMPC Problem 4.3 is the method of computation. The first algorithm requires sequential computation for all local optimizations during one sampling period and the second one only needs parallel computation. This indicates that for the DMPC algorithms, the computation time of the feedback DMPC is independent on the platoon size and can be smaller than it of the open-loop DMPC after the number of vehicles in a platoon is bigger than a critical value, e.g. 9 vehicles under the different prediction horizons.

### 4.6 Conclusion

In this chapter, feedback RMPC Problem 4.1 was applied to the control problem, which led to feedback DeMPC. Then, a distributed MPC algorithm was proposed by reducing the disturbance set and introducing new time varying constraint. It has a larger feasible region in special scenarios because there is a tradeoff between the benefit brought by the reduced disturbance set and the conservativeness introduced by the new constraint. The DMPC also requires extra communication and more computation time compared to the DeMPC. It was shown that the control problem can be solved by both the feedback DeMPC and the DMPC.

In addition, the open-loop MPC algorithms in Chapter 3 were compared with the feedback MPC algorithms.

For the DeMPC schemes, it were found that the feedback algorithm requires much more computational effort than the open-loop one. Their performance is comparable and the open-loop algorithm has a larger feasible region. The advantage of the feedback algorithm is that the real states satisfy the constraints instead of the nominal states.
For the DMPC algorithms, the advantage of the feedback algorithm is that its computational effort does not depend on the size of the platoon. The performance of these two algorithms is similar and the open-loop DMPC has a larger feasible region.
Chapter 5

Application issues

In this chapter, practical issues of vehicle platooning using previous algorithms will be discussed. The content in this chapter aims to serve as the first step towards real application from theoretical development.

5.1 Platoon size

The scaling in Section 3.1 is a theoretical method to deal with strong coupling in terms of disturbance sets in large size. In addition, it also has a large impact on the practical side of vehicle platooning. As described in Section 3.1, it limits the input ranges of all vehicles in the platoon. Starting from the last vehicle which is allowed to use its physical bound of the input, the input bounds of other vehicles are obtained by scaling down the physical bounds backwards through the platoon. Thus, the leader will have the most conservative input range.

The positive side of this method is that the leader has an explicit input range. As shown in [19], the classical controller designed in frequency domain violated the safe distance when the leader applied emergency braking. If the MPC algorithms in this work are used, the leader will have an feasible input bound. When the leader wants to apply an input outside of that bound due to emergency, the system can automatically detect the constraint violation and switch to other systems which ensure safety.

The disadvantage of the scaling is that the number of vehicles in a platoon can be limited. This is because that a minimal input range for safety is required for practice
and shall be a subset of the input bound of the leader. As the platoon size increases, the
input bound of the leader can be more conservative until it equals to the minimal range.
Assuming that the physical input bound is $[-5, 3]$, the scaling matrix $C = \text{diag}(0.9, 0.9)$
and the minimal input bound required for the leader is $[-3, 2]$, the maximum number
of vehicles allowed in a platoon is $5$.

In addition, for the open-loop DMPC, due to the sequential computation required, the
platoon size is also limited by the computation time which will increase along with the
increase in the number of vehicles in the platoon.

## 5.2 Joining and Leaving

As described in Section 3.4, the scenario where one vehicle tries to join or to leave
a platoon can be achieved by distributed synthesis together with other protocols for
initialization. In practice, an even simpler solution is to first quantify the maximum
number of vehicles in future platoon as discussed in Section 5.1, and then to calculate the
feasible input bound and the RCI/RPI set offline for every position in the platoon. Then,
when a vehicle tries to join, based on its order in the platoon, it can be assigned with
pre-computed parameters, which reduces the computation time of distributed synthesis.

## 5.3 String stability

String stability describes the ability of a platoon in attenuating disturbances introduced
by the leader while moving down stream in the platoon. It is an important property in
practice addressed by many literature on vehicle platooning [3] [4] [19]. String stability
is commonly defined and analyzed in frequency domain. Since MPC controllers are
designed in time domain, the definition of string stability in time domain as introduced
in [12] is used in this thesis:

**Definition 5.1.** A vehicle platoon is string stable if for a step change in the velocity
of the leader vehicle $v^1_k$ at time $k = 0$, there exist constant scalars $r^i \in (0, 1)$, $\forall i \in \{2, ..., N_A\}$ such that,

$$\max_{k \geq 0} |a^{i+1}_k| \leq r^i \max_{k \geq 0} |a^i_k|, \quad \forall i \in \{2, ..., N_A\},$$

(5.1)
where \( a_k^i \) is the acceleration of vehicle \( i \) at time \( k \).

The condition (5.1) indicates that when the leader changes its speed, the acceleration response of every vehicle shall not exceed the one of the preceding vehicle. Note that the input \( u^i \) is the acceleration provided by the actuator and \( a^i \) is the actual acceleration of the vehicle. There is usually dynamics between \( a^i \) and \( u^i \) as shown in [12], which is simplified by \( a^i = u^i \) in this thesis.

It was reported that for the platooning control algorithm designed in frequency domain, there is a minimal time gap \( h_{\text{min}}^i \) such that string stability will only hold when \( h^i \geq h_{\text{min}}^i \) [3]. In addition, the value of \( h_{\text{min}}^i \) is smaller for CACC with vehicle-to-vehicle communication than the value for ACC without communication, which implies that the vehicle platoon with CACC can achieve smaller relative distance between 2 vehicles. Thus, an interesting question is that if \( h_{\text{min}}^i \) can also be reduced by moving from decentralized control using RMPC which does not require communication to DMPC with communication.

![Figure 5.1: Input profiles using open-loop DMPC when \( h = 1.4 \)](image)

The 4 MPC algorithms were tested in simulation to find their corresponding \( h_{\text{min}}^i \). An example of the resulting inputs is shown in Figure 5.1 where the open-loop DMPC algorithm is used when there are three vehicles and \( h^i = 1.4 \) for all vehicles. It can be found that in Figure 5.1, the follower reacts to its preceding vehicle with a delay. In
addition, to check string stability, it is important to analyze the time when the leader has a speed change, i.e. the time when $u^1$ goes from zero to a non-zero value. This scenario happens before $time = 100$ where there is a square wave. It can be seen that after $u^1$ goes from zero to a constant value, $u^2$ and $u^3$ start to increase with a delay. However, during the period of increase, $u^3$ is bounded by $u^2$ which is also bounded by the peak value of $u^1$. The peak values of $u^1$, $u^2$ and $u^3$ caused by the step change of $u^1$ are 1.458, 1.458 and 1.457, respectively. Thus, string stability holds for the three-vehicle platoon using open-loop DMPC when $h^i = 1.4$.

Using the method of analysis described above, the result shows that $h_{\text{min}}^i = 1.4$ for all 4 MPC algorithms, which means that for both open-loop MPC and feedback MPC, the moving from DeMPC to DMPC does not influence the value of $h_{\text{min}}^i$. This result of $h_{\text{min}}^i$ under communication from time domain is different from the result from frequency domain. In [3], it was reported that in frequency domain, string stability could be achieved for all $h > 0$ when there is communication without delay. However, even if the communication delay is also not considered in this thesis for the two DMPC schemes, $h_{\text{min}}^i$ cannot be smaller than 1.4 for the DMPC.

In addition, it is also found that $h_{\text{min}}^i$ is independent on the platoon size for all 4 MPC algorithms.

However, note that the model for controller design in this work is simpler than the one used in [3] and the models used for simulation are only double integrators. Thus, the results are suggested to be further studied under more realistic conditions and a detailed study on the frequency-domain analysis of string stability is also recommended.

### 5.4 Choice of the algorithm

Based on the features of the 4 MPC schemes, several recommendations can be made for the choice of algorithm for platooning control.

For vehicle platoons without communication, decentralized control scheme has to be applied. Because the open-loop DeMPC requires less computational effort and has a larger feasible region than the feedback DeMPC, it is recommended to use the open-loop DeMPC.
For platoons with communication, if the platoon is large in size, it is recommended to use feedback DMPC because its computation time does not depend on the platoon size. For platoons in small size, open-loop DMPC can be applied because the total computation time will still be small in this case.
Chapter 6

Conclusions and recommendations

Lastly, conclusions of this work are given and recommendations are also provided for future improvements.

6.1 Conclusions

In this work, starting from two different RMPC algorithms, three problems were solved, including the decentralized control problem of vehicle platooning, distributed control problem of vehicle platooning and the distributed synthesis when the topology of a platoon changes.

First of all, the input bounds were scaled such that the DeMPC schemes can be formulated based on the two RMPC algorithms. For the decentralized control problem, it was shown both DeMPC schemes could solve the control problem. In addition, the open-loop DeMPC has a larger feasible region, less computation time and similar cost compared to the feedback DeMPC.

For the distributed control problem, two DMPC schemes were promoted based on the two DeMPC schemes by establishing communication among vehicles. Most importantly, the proof of recursive feasibility of both DMPC schemes was provided. Then, it was
found that for the two DMPC schemes, the costs were comparable under different prediction horizons and the feasible region of the open-loop one was larger than the feedback one. However, the computation time of feedback DMPC does not depend on the platoon size, which is beneficial for large-scale platoons.

As for the change from the DeMPC to the DMPC scheme, for the open-loop algorithm, the DMPC has larger feasible region than its corresponding DeMPC. However, the price is the higher cost and that sequential computation has to be conducted for the local controllers in the DMPC scheme, which means the total computation depends on the platoon size. For the feedback algorithm, the resulting DMPC scheme can only achieve larger feasible region than the feedback DeMPC in special scenarios. The feedback DMPC also requires larger computational effort because of the communication and online formulation of the parameters. The resulting cost of DMPC is comparable to the one of feedback DeMPC.

For the distributed synthesis, only the algorithms for open-loop DeMPC and open-loop DMPC were provided. Simulation demonstrated that the algorithm in Section 3.4 could synthesize the local controllers automatically when one vehicle tried to join a platoon.

Finally, some application issues of the vehicle platooning with the MPC algorithms were discussed, including the critical value of string stable time gap.

### 6.2 Recommendations

Overall, the future improvements can be in two directions, including the developments on general MPC algorithms and improvements on control of vehicle platooning.

For further development of MPC algorithms, one natural extension of the DeMPC schemes and the DMPC algorithms is to consider general networked systems with weak coupling. The weak coupling means that if the coupling in the networked system is regarded as disturbance, the disturbance set is sufficiently small such that the controller can be synthesized. An example can be found in [5] where decentralized control of the power network system is achieved using a different RMPC algorithm. In addition, the proof of ISS for the two DMPC schemes can be exploited.
For the control of vehicle platooning, more realistic model for controller design should be used as shown in [19]. String stability can also be studied to derive constraints for the MPC controllers such that this property can always be ensured. Finally, the experiments of the algorithms on real vehicles can be conducted to validate the algorithms.
Appendix A

Computation of control invariant sets

A.1 Problem formulation

In the MPC algorithm with communication of [14], the input bound $U_i$ and thus the maximum robust control invariant set $X_i$ are time varying. In this case, the input bound for the leader with the generated upper bound $a_{max}^i$ and lower bound $a_{min}^i$ is shown as

$$U_1 = \left\{ u \in \mathbb{R} \mid \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} u \leq \begin{bmatrix} a_{max}^i \\ a_{min}^i \end{bmatrix} \right\}$$

which is then enlarged to contain the origin. And then the input bounds for the followers is generated as follows

$$U^2(k) = \frac{1}{c} U^1(k - 1)$$

$$U^i(k) = \frac{1}{c} U^{i-1}(k), \quad i \in \{3, ..., N_A\}$$

where $0 < c < 1$ is a design parameter for scaling the input bounds and $N_A$ is the total number of vehicles in the platoon.
The change of input bounds can be regarded as affine transformation as follows

\[ U^1(k - 1) = QU^1(k - 2) \oplus b \]
\[ U^2(k) = PU^1(k - 1) \]
\[ U^i(k) = PU^{i-1}(k), \quad i \in \{3, ..., N_A\} \]

where in the case of [14], \( Q, b \) and \( P \) are all scalars. If considering a more general case where \( u \in \mathbb{R}^m \), it can be obtained that \( Q, P \in \mathbb{R}^{m \times m} \) and \( b \in \mathbb{R}^m \).

Due to the time varying input bounds, \( \mathcal{X}_R^i \) is also time varying which cannot be computed efficiently online. Given \( U_{\text{new}} = QU \oplus b \), a state constraint \( \mathcal{X} \) and a linear time invariant model \( x(k + 1) = Ax(k) + Bu(k) + w \), the goal of this report is to investigate if there is a relationship between \( \mathcal{X}_R(U) \) and \( \mathcal{X}_R(U_{\text{new}}) \). If this relationship exists, the maximum robust control invariant set \( \mathcal{X}_R \) could be adapted online to the affine transformation of the input bound, which facilitates the online implementation of the MPC algorithm with communication. To simplify the problem, only model without disturbance and corresponding maximum control invariant set \( \mathcal{X}_V \) are considered.

Given an input bound, \( \mathcal{X}_V \) is calculated by iterating the computation of the one-step set \( Q(\bullet) \) as follows

\[ K_0(\mathcal{X}, \mathcal{X}) = \mathcal{X} \]
\[ K_i(\mathcal{X}, \mathcal{X}) = Q(K_{i-1}) \cap \mathcal{X}, \quad i \in \{1, ..\} \]

where \( K_i(\bullet, \bullet) \) is a \( i \)-step controllable set. This iteration terminates when \( K_i = K_{i+1} \), which does not always happen, and then \( \mathcal{X}_V = K_i \). Based on the iteration, it can be found the computation of control invariant set requires algorithms for the computation of one-step set, the intersection of two sets and the equal set testing to check if the iteration should be terminated. In addition, an algorithm for the affine transformation of a set is also required to formulate a new input bound.

The intersection and equal set testing have been addressed in [1] which will not be discussed in detail. The main focus of this report will be on the affine transformation and the computation of one-step set.
A.2 Simplified case

In this section, to simply the problem, only linear transformation $QU$ and the relationship between $Q(X)$ with $U$ and $Q_{\text{new}}(X)$ with $QU$ are investigated.

A.2.1 Invertible $Q$ and $A$

Firstly, $Q \in \mathbb{R}^{m \times m}$ and $A \in \mathbb{R}^{n \times n}$ are assumed to be invertible. The general input bound and the state bound are represented by finite number of linear inequalities as follows

$$U = \{ u \in \mathbb{R}^m | Gu \leq g \}, \quad X = \{ x \in \mathbb{R}^n | Sx \leq s \}$$

where $G \in \mathbb{R}^{M \times m}$, $g \in \mathbb{R}^M$, $S \in \mathbb{R}^{N \times n}$ and $s \in \mathbb{R}^N$. Then the new bound $QU$ is shown as follows

$$QU = \{ u \in \mathbb{R}^m | GQ^{-1}u \leq g \}$$

The one-step set $Q$ and $Q_{\text{new}}$ are shown as follows

$$Q(X) = \{ x(k) \in \mathbb{R}^n | \exists u(k) \in U : Ax(k) + Bu(k) \in X \}$$

$$Q_{\text{new}}(X) = \{ x(k) \in \mathbb{R}^n | \exists u(k) \in QU : Ax(k) + Bu(k) \in X \}$$

The one-step sets can be calculated via projection operation or Minkowski sum.

For projection operation, new sets $H$ and $H_{\text{new}}$ can be constructed as follows

$$H = \left\{ \begin{bmatrix} x \\ u \end{bmatrix} \in \mathbb{R}^{n+m} \left| \begin{bmatrix} SA & SB \\ 0 & G \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} \leq \begin{bmatrix} s \\ g \end{bmatrix} \right. \right\} \quad (A.1a)$$

$$H_{\text{new}} = \left\{ \begin{bmatrix} x \\ u \end{bmatrix} \in \mathbb{R}^{n+m} \left| \begin{bmatrix} SA & SB \\ 0 & GQ^{-1} \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} \leq \begin{bmatrix} s \\ g \end{bmatrix} \right. \right\} \quad (A.1b)$$

Then the one-step sets can be obtained by projecting $H$ and $H_{\text{new}}$ to obtain the first $n$ coordinates.
For the method to compute the one-steps using Minkowski sum, since $x(k) = A^{-1}(x(k + 1) - Bu(k)) \in Q(X)$ if $A$ is invertible, the one-step sets can be represented as follows

\[ Q(X) = A^{-1}(X \oplus (-BU)) \]
\[ Q_{\text{new}}(X) = A^{-1}(X \oplus (-BQU)) \]

where

\[ X \oplus (-BU) \triangleq \{ \phi \in \mathbb{R}^n | \phi = x + \psi, \exists x \in X, \psi \in (-BU) \} \quad (A.2a) \]
\[ X \oplus (-BQU) \triangleq \{ \phi \in \mathbb{R}^n | \phi = x + \psi, \exists x \in X, \psi \in (-BQU) \} \quad (A.2b) \]

Now the problem becomes how to calculate the Minkowski sum and what the relationship is between (A.2a) and (A.2b). The Minkowski sum can be computed by finding the vertices of the corresponding sets and constructing the convex hull of the sum of the vertices, which is shown as follows

\[ X \oplus (-BU) = \text{conv}\{ \phi \in \mathbb{R}^n | x \in \text{vert}(X), \psi \in \text{vert}(-BU), \phi = x + \psi \} \]
\[ X \oplus (-BQU) = \text{conv}\{ \phi \in \mathbb{R}^n | x \in \text{vert}(X), \psi \in \text{vert}(-BQU), \phi = x + \psi \} \]

based on which, the one-step sets can be calculated as follows

\[ Q(X) = \text{conv}\{ \phi \in \mathbb{R}^n | x \in \text{vert}(X), \psi \in \text{vert}(U), \phi = A^{-1}x - A^{-1}B\psi \} \quad (A.3a) \]
\[ Q_{\text{new}}(X) = \text{conv}\{ \phi \in \mathbb{R}^n | x \in \text{vert}(X), \psi \in \text{vert}(U), \phi = A^{-1}x - A^{-1}BQ\psi \} \quad (A.3b) \]

The next question is that the relationship between $Q(X)$ and $Q_{\text{new}}(X)$ based on (A.1) and (A.3).
Bibliography


