Universidad de Sevilla

Departmento de Sistemas y Automática

Doctoral Thesis

Distributed Model Predictive Control Based on Game Theory

Author:
José María Maestre Torreblanca

Supervisors:
Dr. Eduardo Fernández Camacho
Dr. David Muñoz de la Peña Sequedo

October 22, 2010

Los problemas de los demás son a menudo la mitad de tus soluciones

La Buena Suerte
Alex Rovira Celma y Fernando Tras de Bes

The problems of others often hide half of the solution to your problems

The Good Luck
Alex Rovira Celma y Fernando Tras de Bes

Acknowledgements

In first place, I would like to thank my family, particularly my parents. I hope they will be glad to know that there is going to be another doctor carrying their surnames.

I would also like to thank my *academic* parents, the doctors Eduardo Fernández Camacho and David Muñoz de la Peña. They have been very supportive during these years and have taught me how to become a true researcher. I appreciate all their confidence and the countless hours they have spent working with me.

There have been many other researchers that have helped me with this work. In particular, I would like to recognize the role of professor Anders Rantzer, who granted me the possibility of working seven months with his team in Sweden.

I also want to thank all my colleagues in Seville. Working in the Systems and Automation Department would not be such a privilege without people like you.

At last, but not least, I would like to thank my girlfriend, Isabel, for all the time I have not been able to spend with her because of this thesis you have in your hands.

Contents

1 Distributed Model Predictive Control and Game Theory **1**

1.1 Sensors and actuators networks 2

1.2 Distributed control 3

1.2.1 Types of control 3

1.2.2 Types of neighborhood 4

1.2.3 Types of communication 5

1.2.4 Types of variable update 5

1.2.5 Types of agent 6

1.3 Model predictive control 6

1.4 Distributed model predictive control 9

1.5 Game theory 10

1.5.1 Non-cooperative game theory 11

1.5.2 Cooperative game theory 14

1.6 Literature review 15

1.6.1 Decentralized schemes 16

1.6.2 DMPC based on information broadcast 17

1.6.3 DMPC based on agent collaboration 18

1.7 Objectives of the thesis 22

1.8 Thesis outline 23

1.9 Contributions 24

2 Distributed Model Predictive Control Based on Game Theory for Two Agents 27

2.1 Problem formulation 28

2.2 Stability properties 36

2.2.1 Design procedure 39

2.3 Simulation examples 41

2.3.1 Two double integrators with coupled inputs 42

2.3.2 Application to a supply chain problem 44

2.4 Robustness of the proposed approach against data losses 54

2.5 The four tank process 59

2.5.1 The four tank process 60

2.5.2 Four tank plant model 63

2.5.3 The benchmark problem 65

2.5.4 Controllers under test 67

2.6 Conclusions 73

3 Distributed Model Predictive Control Based on Game Theory for Multiple Agents 75

3.1 Problem formulation 76

3.2 Proposed DMPC controller 78

3.3 Stability 83

3.4 Controller design procedure 86

3.5 Example 94

3.6 Application to a supply chain problem 98

3.7 Application to control of irrigation canals 103

3.8 Conclusions 107

4 Distributed Receding Horizon Kalman Filter 109

4.1 Problem formulation 110

4.2 Dual decomposition 114

4.2.1 Coordination alternatives for the price update 117

4.3 Examples 119

4.3.1 Application to mobile robot localization 120

4.3.2 Application to traffic and speed control 122

4.4 Conclusions 123

5 Applications of Cooperative Game Theory to the Control and Estimation of Distributed Systems 127

5.1 Cooperative games 129

5.2 Distributed control problem formulation 132

5.2.1 Distributed control algorithm 133

5.2.2 Network modes 135

5.2.3 Link analysis 136

5.2.4 Agent analysis 136

5.2.5 Coalitional game design method 137

5.3 Distributed control simulation results 140

5.4 Distributed estimation problem formulation 150

5.4.1 Network modes 152

5.4.2 Link analysis 152

5.4.3 Agent analysis 153

5.4.4 Coalitional game design method 153

5.5 Distributed estimation simulation results 155

5.6 Conclusions 160

6 Conclusions and Future Research 161

6.1 Conclusions 162

6.2 Future research 163

A Resumen en castellano **167**

A.1 Redes de sensores y actuadores 169

A.2 La teoría de los juegos 170

A.3 Objetivos de la tesis 173

A.4 Estructura de la presente tesis doctoral 174

A.5 Contribuciones al estado del arte 175

Chapter 1

Distributed Model Predictive Control and Game Theory

Traditionally, control theory has coped with information and timing constraints in a centralized fashion. The design of control architectures is made assuming that all the information is available at a single point at the right time. There is no doubt that, if this assumption holds, the best possible control performance can be achieved. Unfortunately, centralized architectures can not always be used in practice. There are different factors that hinder the application of these schemes. In first place, real systems may not have a model that capture correctly their dynamics. Moreover, even if a model can be obtained, it may be too complex to be useful to design a controller. Likewise, there are other important limitations that may make impossible the use of a centralized architecture. For example, the system may be geographically disperse, being impossible to gather all the information at a single point at the right time. Other times it is a matter of privacy: the subsystems that compose the overall system may be independent and may have incentives to keep some information secret. This could be, for example, the case of a supply chain.

When one of the referred situations appears, it is not possible to use a centralized controller. It is at this point where decentralized and distributed controllers come into play. The idea behind these schemes is simple: the centralized problem is divided in several different parts whose control is assigned to a certain number of local controllers or *agents*. Therefore, each agent does not have a global vision of the problem. This is probably the main feature that characterizes decentralized and distributed systems. Depending on the degree of interaction that exists between the local subsystems, the agents may need to communicate so that they can coordinate themselves. If communication is needed, we speak of distributed systems. By contrast, when the degree of interaction is low enough and agents can afford

their control tasks with no communication between them, we speak of decentralized control systems.

Decentralized and distributed schemes have important advantages that justify their use. The first advantage is that in general these schemes are easier to implement. Their computational requirements are lower because a difficult problem is substituted by several smaller problems. In addition, these schemes are scalable. Their inherent modularity simplifies the system maintenance and the possible expansions in the control system. Moreover, the modularity provides robustness in comparison with a centralized controller. A possible failure does not have to affect the overall system. For this reason, decentralized and distributed systems have a greater tolerance to failures. Nevertheless, these systems have also several drawbacks that have to be taken into account, being the main one the loss of performance in comparison with a centralized controller. This loss depends on the degree of interaction between the local subsystems and the coordination mechanisms between the agents.

This thesis focuses on the development of distributed control and estimation techniques with low communicational burden and on the analysis of the properties of a given distributed scheme. These objectives are developed in a distributed model predictive control framework using tools from game theory. In distributed applications, the closed-loop performance is the result of a trade-off between the number of communications made by the agents and the costs of the communication itself. It is frequent to see in the literature that costless communication is assumed, which is a dangerous assumption. Actually, yet in 1992 Peter Deutsch pointed out the risks of assuming costless communication in his list of fallacies of distributed computing [21]. Communication is costly in several dimensions, specially in terms of time and energy consumption, which may be critical factors for may applications.

In this chapter we present some background for the research addressed in this thesis and review the most relevant results that can be found in this field. Section 1.1 presents an introduction to sensor and actuator networks. Next, section 1.2 provides some basic concepts and taxonomies of distributed control. In section 1.3 we describe the basics of model predictive control. Section 1.4 deals with the distribution of the centralized control problem among a set of agents. Section 1.6 introduces some useful taxonomies for the distributed control problem. Finally, section 1.6 surveys the most important DMPC algorithms that can be found in the literature.

1.1 Sensors and actuators networks

Decentralized and distributed systems have been a subject of study for a long time, but it has not been until the last decade when they have been at their very peak. The renewed interest

in distributed and decentralized schemes has been mainly motivated by the proliferation of low cost wireless transceivers and their wide range of applications. Wireless autonomous networks provide a mean to measure or actuate much cheaper than the traditional wired solutions. While the first wireless solutions were expensive and had a small autonomy, the new developments have brought devices with years of battery life at a low price.

Probably, the most relevant technologies in this sense are related with the IEEE 802.15.4 standard, a protocol designed for wireless personal area networks (WPAN). In contrast with other technologies such as Wifi or Bluetooth, which are oriented to high bandwidth applications, 802.15.4 solutions aim to optimize the battery lifetime. For this reason, it offers a relatively small bandwidth, enough to satisfy the communicational needs of many control applications. This factor, together with the ease of deployment of wireless networks, explains the great proliferation of distributed applications.

Without any doubt, these networks will change the world as we know it at industrial and home level. For example, paradigms such as pervasive computing were just an utopia a few years ago and now seem to be perfectly possible. However, there are may questions that have to be addressed in order to optimize the use of these new technologies such as the analysis and synthesis of distributed systems, the integration of heterogeneous technologies in a same network or the optimization of the information transmitted through the network, to name a few.

1.2 Distributed control

In this section we present some fundamental taxonomies that allow to classify the schemes that have been presented in the literature. Some of this classifications are not new and can be found in [13] and [71].

1.2.1 Types of control

The controllers can be classified as a function of how many agents participate in the solution of the control problem and the relative importance between them. We say that a control system is centralized if there is a single controller that solves the plantwide problem. The control is decentralized when there are local controllers in charge of the local subsystems of the plant that require no communication among them. When the local controllers communicate in order to find a cooperative solution for the overall control problem the control system is distributed. Finally, if there are different control layers coordinated to take care of the process

the control system is hierarchical. In this case, upper layers manage the global objectives of the process and provide references for the lower layers, which control directly the plant.

In this thesis we will focus mainly, but not exclusively, on distributed controllers. Note that, as we stated before, the performance of the closed-loop system depends on the decisions that all the agents take, so cooperation and communication policies become very important issues.

At this point it is convenient to remark that the models used to design the control system have to be coherent with the type of control adopted. For this reason, models can also be classified in a similar way.

1.2.2 Types of neighborhood

In a distributed control system, the state and control actions of a given subsystem may affect other subsystems. Different types of interaction between subsystems can be defined based on the relative degree of interaction between the inputs and states of each subsystem. We speak of an interaction of type zero when two different agents share a state variable or control directly the same manipulated variable. Interaction of type one is referred to the case in which the coupling between the variables controlled by two different agents is strong enough to require communication and coordination between them. In other words, it is not possible to consider the interaction as a mere disturbance. This is the most frequent type of neighborhood in distributed systems. This classification can be extended to a general case. For example, a type two would correspond to the case in which the interaction of a given agent i over an agent j is strong enough to induce a considerable disturbance on a third agent k, which in general will by a type 1 neighbor of agent j.

Note that interactions and neighborhood may depend on the way the centralized control problem is distributed between the agents. It is beyond the scope of this work to provide a mechanism to break the centralized models into subsystems. Nevertheless there are works on the literature that deal with this problem, see for example [67, 32].

In this thesis we will deal with the types of neighborhood zero and one. We will assume that the disturbances due to other types of interactions are neglectful.

1.2.3 Types of communication

Communication plays an essential role in distributed control. It can be classified in several classes according to different factors related with the protocol that is followed during the communication. In first place, it is possible to distinguish between synchronous and asynchronous communication depending whether there is or not a strict timing in the communication process that determines when an agent can communicate. Second, we can classify the communication as serial or parallel depending on whether one or more than one agent is allowed to communicate at the same time. Finally, we can classify the communication taking into account the number of communications employed. This is an important factor for some systems. It is needless to say that decentralized algorithms have no communication steps. With one communication step it is possible to establish a distributed control scheme in which the agents can inform their neighbors about their plans so that this information considered in the optimization procedure. A consensus or agreement between the agents is only possible if two or more communication steps are used. The number of communications is finite only if the negotiation process converge. However note that it is possible that the number of iterations required to converge is greater than the maximum number of communications allowed.

The algorithms that we have developed in this work are designed to be implemented in parallel. Naturally, these algorithms also admit a serial implementation. In general we require synchronous communication although asynchronous communication can be used in our multiple agent distributed control scheme. Finally, the number of communications that is used by the agents is low taking into account that all the decisions are cooperative. For two agents we have developed an scheme with two communication steps and for multiple agents we have proposed an algorithm that offers a good performance with an average number of five communications per agent.

1.2.4 Types of variable update

Agents may update their manipulated variables in several ways. It is possible to establish a classification as a function of the way in which the agents update their variables from a social point of view. According to this, we can classify the variable update as exclusive, if each agent update its variables autonomously, shared, if an agent allows other agents to manipulate its variables, or democratical, if an agent takes into account the suggestions of other agents when he calculates the new values of its manipulated variables.

The type of variable update that we have considered in this thesis is the democratical one. The decisions are taken in a social way whenever it is possible.

1.2.5 Types of agent

The agents are the essence of distributed control systems. There are important features that allow us to classify them according to different parameters. In first place, we can classify them according to the amount of information each agent has about the whole system. This information has a great impact in the distributed control algorithm used. We say an agent is blind if the agent has only information about its corresponding local subsystem. In other words, the agent ignores the rest of subsystems and their influence is modeled as a disturbance. This is the case, for example, of decentralized control systems. An agent is said to be a standard agent if it knows his own dynamics and also how other agents affect him. This knowledge is the base of a possible negotiation procedure with other agents. An agent is samaritan if it is an standard agent that in addition knows how it affects the other agents as well. This information allows the agent to choose his control actions minimizing the possible negative effects on its neighbors. Finally, an agent is omniscient if it has full centralized information or at least it knows all the information it needs about the system. In second place we can classify the agents according with their attitude, which is another important factor. In this context, attitude is related with the will of collaboration between the agents. In this sense the agent's attitude is noncooperative if the agent behaves selfishly, that is, it only tries to maximize its own utility function (i.e., to minimize its cost function). On the other hand, the agent's attitude is cooperative when the agent tries to minimize not only its cost but the cost of its neighbors. Thus, the agent can make a sacrifice in terms of his own welfare to help the system reach a better global situation.

One of the main assumptions in our control schemes is to consider standard agents from the informational point of view. In addition, the agents are considered to have a cooperative attitude.

1.3 Model predictive control

Although there are numerous different control techniques, in this thesis we will focus on Model Predictive Control (MPC). MPC, also known as receding horizon control (RHC), is a popular control strategy for the design of high performance model-based process control systems because of its ability to handle multi-variable interactions, constraints on control (manipulated) inputs and system states, and optimization requirements in a systematic manner. MPC takes advantage of a system model to predict its future evolution starting from the current system state along a given prediction horizon. Nominal MPC controllers consider discrete models of the following form:

$$x(t+1) = f(x(t), u(t))$$
$$y(t) = g(x(t), u(t))$$

where t represents the sample time and $x(t) \in \mathbb{R}^n$, $u(t) \in \mathbb{R}^m$ and $y(t) \in \mathbb{R}^q$ are the state and input vectors of the system respectively. The model is used recursively at time t to predict the state or outputs in a finite horizon of length N when a given input trajectory $U(t) = \{u(t \mid t), u(t+1 \mid t), \ldots, u(t+N-1 \mid t)\}$ is applied. The resulting state at the end of the horizon, $x(t+N \mid t)$, is called terminal state. The mathematical properties of the functions f and g depend on the type of dynamics that are modeled (for example linear, hybrid, nonlinear...). The class of models considered by a given MPC defines the properties of the controller as well as the complexity of the resulting optimization problem. In this thesis we will focus on linear models, although some of the results can be extended to more general classes of systems.

One of the most appealing features of MPC is its ability to handle the constraints on the values of the states and inputs in an explicit way. Real processes have limits in the values of all their variables. For example, a valve cannot be opened negatively. Mathematically, this is modeled by defining a set of admitted values for the state, output and input variables, for example:

$$\begin{array}{rcl} x(t) & \in & \mathcal{X} \\ u(t) & \in & \mathcal{U} \end{array} \tag{1.1}$$

where $\mathcal{X}$ and $\mathcal{U}$ are the sets that define the admissible values for the state and input variables.

The objective of model predictive control is to minimize a given performance index that depends on the future predictions of the state, output and input variables. The performance index is a cost function which defines an optimization criterium that is used to determine which control action sequence offers a better performance. Mathematically it is function that expresses the cost associated to a certain evolution of the system in the horizon interval considered. A typical form of the cost function is

$$J(x(t), U(t) = \sum_{j=0}^{N-1} L(x(t+j \mid t), u(t+j \mid t)) + V(x(t+N \mid t)) \tag{1.2}$$

where $L(\cdot, \cdot)$ represents the stage cost of the system at time $t+j$ and $V(\cdot)$ is the terminal cost. Note that the cost function depends on the current state $x(t)$ of the system and the sequence of *possible* control actions $U(t)$. Therefore, it can be used as a minimization criterium to choose

the best possible control sequence for the system. Moreover, the minimization procedure has to include the constraints. This can be seen as if cost function *punished* those control sequences that result in a violation of any of the constraints. When using continuous time models MPC optimizes over a family of piecewise constant trajectories with a fixed sampling time and a finite prediction horizon so that a finite dimensional optimization problem is obtained.

At each sampling time, the function (1.2) is minimized in order to find the optimal control sequence $U^*(t)$. Once the optimization problem is solved, only the first manipulated input value is implemented and the rest of the trajectory is discarded; this optimization procedure is then repeated in the next sampling step. This is the so-called sliding or receding horizon scheme.

Once the elements that characterize an MPC controller have been defined, it is possible to summarize the control algorithm followed by this technique:

1. Measure the current state of the system $x(t)$.
2. Calculate which actions provide the best performance over the horizon by solving the following optimization problem:

$$\begin{aligned} &\min_{U(t)} J(x(t), U(t)) \\ &s.t. \\ &x(t+j \mid t) \in \mathcal{X},\ \forall j \in [1, N] \\ &u(t+j \mid t) \in \mathcal{U},\ \forall j \in [0, N-1] \end{aligned} \tag{1.3}$$

3. Apply the optimal inputs calculated for the first time sample of the prediction horizon, that is, $u(t \mid t)$ and return to step 1.

The success of MPC in industrial applications [12] has motivated an important amount of research on the stability, robustness and optimality of model predictive controllers. This success is logical given the advantages of MPC in comparison with its drawbacks. In the positive side we have that MPC is easy to tune if an appropriate model is available. MPC also works very well with control problems that are difficult to solve with other control techniques. For example, it deals naturally with multiple inputs and multiple outputs systems, constraints, delays and disturbances. On the other hand, there are disadvantages that make the implementation of MPC difficult on some systems. Possibly, the main drawback of MPC has to do with its strong computational requirements. Factor as nonlinearities or constraints may make MPC unsuitable for systems with fast dynamics. In particular, nonlinearities lead to non convex optimization problems, which in general are very hard to solve . Likewise, the

performance of MPC heavily depends on the quality of the model used. The worse the model is in comparison with the real system, the poorer results are obtained.

As it has been seen, the cost function and the system model have a great impact in the difficulty of the MPC optimization problem. In this thesis we use linear models and quadratic functionals, which allow to solve the optimization problem using quadratic programming (QP). The existing methods for QP problems are very efficient and allow us to solve the optimization problem on line. Moreover, in the case that there is no constraints, the optimization problem can be solved explicitly.

1.4 Distributed model predictive control

The drawbacks of MPC hinder its application to large-scale systems. Typical examples of large scale systems are transportation systems such us traffic, water or power networks [78]. In these systems, the computational requirements or the impossibility of obtaining a centralized model are major problems that MPC cannot avoid. Besides large-scale systems, it is also difficult to apply centralized MPC to networked systems; the distributed nature of these systems require control schemes that do not depend on any centralized element. During the last years, we have assisted to the proliferation of networked control systems (NCS) which have emerged from the augmentation of the dedicated local control networks with additional networked (wired and/or wireless) actuator/sensor devices, which have become cheap and easy-to-install [101] [73].

In practice, most large scale and networked control systems are based on a decentralized architecture; that is, the system is divided into several subsystems, each controlled by a different agent which may or may not share information with the rest. Each of the agents implements a controller based on a reduced model of the system and on partial state information, which in general results in an optimization problem with a lower computational burden. As it was shown in the list of taxonomies of section 1.2, different possibilities arise at this point depending on several factors. In first place, the degree of centralized information shared by the agents is important. In general, the more information the agents have, the better control performance can be obtained. However, the amount of information shared by the different agents may have a cost that sometimes outweighs the improvement in the performance. Other issues such as scalability also have to be taken into account when defining access to the model and the state of the system. In second place, the way the centralized cost function is separated between the agents, that is, the way the local cost functions are formed, has relevance too. This factor is related too with the agent's attitude, which is another important factor that defines the distributed control scheme. Agents can be cooperative or selfish. Local controllers may not be willing to cooperate if cooperation implies that they

have to sacrifice their local performance. Finally, the number of communications that the agents can make in order to take a decision is also a relevant factor. Communication requires time and energy, which may be design limiting factors.

These factors generate a family of different distributed MPC problems which demand different solutions adapted to the particularities of each concrete problem. For these reason, different DMPC algorithms have emerged in the literature. Later in the chapter the most important ones will be surveyed. Now it is preferable to see the role that game theory plays in DMPC.

1.5 Game theory

Game theory is a mathematical field that studies the process of interactive decision making, that is, situations in which there are several entities, namely *players* or *agents*, whose individual decisions determine jointly the final outcome. In other words, game theory translates into mathematical models situations of conflict and/or cooperation between rational and intelligent agents. Given that conflict is present everywhere in the world, its results have been applied to a great variety of fields such as biology, sociology, politics, engineering or economy, to cite just a few examples. The origins of this mathematical field date back to 1921, with the publication of several papers about "la théorie du jeu" by the French mathematician Émile Borel. Some years later, in 1928, John Von Neumann published the article "Theory of Parlor Games", with important contributions that gave the emerging field mathematical respectability [77]. Finally, it is worthy to mention that in 1944 Von Neumann published together with the economist O. Morgenstern the book *Theory of Games and Economic Behavior* [100], an influential and pioneering work that showed the potential contribution of game theory to economics.

The need of game theory tools depends of the degree of interaction between the agents. Interaction is actually a key concept in this context. It is hard to find a situation in which there is no interaction with other entities that induce at least a degree of uncertainty in the decision making process. If this uncertainty is small enough then individual decision making techniques such as classical optimization can be used. For example, the problem of how many goods has to produce a firm in order to maximize its own profit may be seen as an individual decision making problem, but this is just a mere simplification because there are other factors that have more or less influence in the firm's profit: price of competence products, substitutive goods, or just the general situation of the economy.

In the context of game theory, some common words acquire a special meaning; they are still close to their ordinary meaning in most cases but there are some subtleties that cannot

be ignored. Next, some of these terms will be introduced and defined:

- **Game**. Situation whose outcome depends on the individual decision of several entities or players.
- **Players**. Each of the two or more entities whose decisions determine the outcome of the game. In this text players will be called indistinctively agents. A player will be said to be rational if he takes his decisions consistently to his objectives, that will be expressed as the optimization of a certain utility function. A player will be said to be intelligent if he knows everything that can be known about the game and can make inferences from that information. For example, in economics agents are assumed to be rational, but not intelligent because a complete understanding of the whole economic model from players is not expected.
- **Actions**. Possible choices a player can make when is his turn to decide.
- **Strategies**. Complete plan of actions of a player for the game, that is, a strategy determines which action must be chosen for a player at any time he has to decide. A profile of strategies is a set of possible strategies for a given player.
- **Outcome**. One of the possible endings or results for the game. Each outcome implies consequences for the players.
- **Payoff**. What is given to a player at the ending of the game, that is, the utility the player attributes to that particular ending.

Games can be classified in different categories. The most meaningful one classifies games as cooperative or non-cooperative. Again, the term *cooperative* has a special meaning in this context. A game is said to be cooperative if agents can negotiate among them and commit themselves to follow common binding strategies. If the agents cannot negotiate, the game is said to be *non-cooperative* because the agents have no guarantees about the behavior that the rest of the agents will have in the present and the future. The most important and famous results of game theory emerged in non-cooperative game theory. Nevertheless, cooperative game theory provides mathematical tools that suit very well for distributed control problems. For this reason we will review briefly these two subfields.

1.5.1 Non-cooperative game theory

Von Neumann and Morgenstern only studied two person non-cooperative games, which are trivial since it is impossible to talk about coalitions in this case. In the case of games with

more than two agents, they focused on the possible coalitions that players would establish. N-person games were not studied from a non-cooperative perspective until the 1940s and 1950s, when the mathematician John F. Nash extended game theory in that direction [68]. In this subsection we will introduce some of the results of noncooperative game theory that have been used in this thesis. The reader interested in a deeper insight on the topic is recommended to see some of the classical references in the literature such as [9, 66].

The most common way to represent a game is the strategic or normal form. This representation is static in the sense that the time instants in which the decisions are taken are not important an it can be assumed that all the agents make their decisions simultaneously. Mathematically, a game Γ in strategic form is given by:

$$\Gamma = (M, (U_i)_{i \in M}, (J_i)_{i \in M}) \tag{1.4}$$

where M is a nonempty set of agents, U_i is a nonempty set of strategies for agent i and $(J_i)_{i \in M}$ is a payoff function that maps the set of possible strategies chosen by the players in the set of reals, specifying the payoff for every agent.

For simplicity we will use a 2-person game to introduce the most relevant concepts of noncooperative game theory. In this case, it is possible to pose the strategic form of the game as a table. A well known example that can be found in the literature example is the prisoner's dilemma, which is presented next [77]:

Two members of a criminal gang are arrested and imprisoned. Each prisoner is in solitary confinement with no means of speaking to or exchanging messages with the other. The police admit they don't have enough evidence to convict the pair on the principal charge. They plan to sentence both to a year in prison on a lesser charge. Simultaneously, the police offer each prisoner a Faustian bargain. If he testifies against hist partner, he will go free while the partner will get three years in prison on the main charge. Oh, yes, there is a catch... If both prisoners testify against each other, both will be sentenced to two years in jail.

	B refuses deal	B turns state's evidence
A refuses deal	1 year, 1 year	3 years, 0 years
A turns state's evidence	0 years, 3 years	2 years, 2 years

As it can be seen in the table, there are four possible outcomes for this game. Each player has two possible strategies, accept or refuse the deal. However, given that the game is symmetric, if both players are intelligent and rational, only one of the two outcomes in the

diagonal will be the final outcome of the game. Each of this solutions has an special meaning in game theory.

The first possible outcome that we will analyze is the case in which both player turn state's evidence. This situation corresponds to the famous Nash equilibrium, which roughly speaking can be defined as the no-regrets outcome, that is, none of the players is unhappy with his decisions during the game once the game has ended. Mathematically a Nash equilibrium is defined as a strategy profile U^* in which no unilateral deviation in strategy by any single player i is profitable for that player, that is,

$$\forall i, u_i \in U_i, u_i \neq u_i^* : J_i(u_i^*, u_{-i}^*) \geq J_i(u_i, u_{-i}^*)$$

where u_i and u_i^* are different strategies carried out by the player i and u_{-i}^* stands for the strategy implemented by the rest of agents in the Nash equilibrium. As it can be seen, in this outcome none of the players regrets his choice. For example, let us suppose that A would have choice to refuse the deal, then A would have got the worst possible outcome for his interests. The same analysis can be made by player B. For this reason, both are equally *happy* with the decisions they made, although, paradoxically, they would have been better if both of them have refused the deal.

The other possible logical outcome of the game is the social optimum, which happens when the two players cooperate and refuse the deal. This is related with the concept of Pareto efficiency. Roughly speaking, Pareto optimality has to do with an outcome in which no agent can be made better off without making worse off at least other agent. Mathematically, a payoff vector $v = (v_i)_{i \in M})$ is weakly efficient in the sense of Pareto if there is no other payoff vector $w = (w_i)_{i \in M})$ such that $w_i \geq v_i \forall i \in M$. In addition, if the inequality holds strictly, that is $w_i > v_i \forall i \in M$, then v is strongly efficient in the sense of Pareto. It can be seen that the payoffs that are Pareto efficient are maximals of the set of possible outcomes.

It is important to remark one of the most important features of noncooperative games: the logical outcome of a prisoner's dilemma played by selfish players is its Nash equilibrium, which paradoxically is worse than the Pareto efficient outcome. To avoid this paradox, it is necessary to introduce some kind of social concern in the payoffs of the players, so that they are not only interested in their own welfare.

A distributed control problem can be usually posed as a game of the form 1.4. The process is not straight-forward, though. There is a set of agents whose strategies or control actions have side effects in the costs of the rest of the agents. Nevertheless, defining the cost functions of the agents may not be an easy task. This basic situation may be enhanced by,

for example, letting the players iterate so that they react to each other's control actions until convergence to a Nash equilibrium has been obtained (in case that the interactions converge, which may not be possible). Anyway the concepts that we have seen in this section still prevail in these more sophisticated scenarios. As it can be seen, Pareto efficiency and Nash equilibria are important in the analysis and design of distributed control systems. In fact, all these concepts have become popular in the study of distributed control schemes and they can be found frequently in the literature, see for example [9, 43, 98, 27].

1.5.2 Cooperative game theory

Cooperative game theory studies situations of mutual interaction between a set of agents which can negotiate among them and commit themselves to follow common binding strategies. As a result of the bargaining process, the set of agents might be divided into several subsets that are called coalitions. The role of game theory in this field is to study which coalitions of agents should be formed and to analyze how the cost or benefits from cooperation should be distributed between the members of a coalition. Note that the existence of a communication channel is implicit in this branch of game theory. In this subsection some concepts of cooperative game theory that have been used in this thesis will be introduced. The reader interested in a deeper insight on the topic is recommended to see [47].

In its most basic form, a cooperative game is defined only with two elements, a set of different *players* and a function that assigns a value to each of the possible coalitions of players. In this point we have to remark that the value of the coalition represents the cost to reach the common goal without the assistance of the agents that are not present in the coalition. Nevertheless, there are other elements that define the class of cooperative games in which we are interested, for example the network. The study of the influence of the network in cooperative game theory began decades ago with the work of Myerson [65]. The necessary and sufficient condition for any two agents to communicate, and hence cooperate, is that they are at least indirectly connected by the network, that is, there exists a path of active links that connect them. In addition, a cooperative game can also take into account the costs of communication. Therefore, it can be considered that the existence of each link has a fixed cost associated to its use. With all these ingredients, we can define a cost-extended communication situation. Cost-extended communication situations allows one to study several inherent properties of the agents and the network [47] and are applied in this thesis as an analytical tool for networked control systems .

In general, given a game there are several possible rules to determine a payoff vector as a solution for the players in the game. The most popular rule is the Shapley value [66], which is the only allocation rule $\gamma(N, v)$ that verifies the following axioms: efficiency, additivity,

symmetry and passive player property. The Shapley value can be interpreted as the payoff vector that gives to each player his expected marginal contribution to a random coalition.

Note that despite the solutions in cooperative games are focused in the obtention of payoff vectors to estimate the distribution of costs or benefits between the players, in this thesis it will be shown that it is possible to use these values as tools for the analysis of relevance of the agents and the links in a distributed control problem.

1.6 Literature review

During the last few years there has been a great interest in the research of distributed control systems. In particular, several distributed MPC schemes have been proposed in the literature that deal with the coordination of separate MPC controllers that strive to obtain optimal input trajectories in a distributed manner. In this section we provide a review of the most important contributions that can be found in the literature in this area. Other reviews of this topic can be found in [84] or [88]. In addition, in [13] basic collaboration algorithms are provided with an extensive list of conditions to ensure convergence and stability.

The main goal of decentralized and distributed algorithms is the same: obtaining the best possible solution for the problem 1.3 in a distributed fashion. Nevertheless, the variety of distributed control problems demands different solutions able to adapt to the particularities of each problem. In order to simplify the mathematical presentation of the different algorithms, we will adopt a simplified notation. In particular we will group in a single vector S all the variables involved in the optimization problem and we will omit the dependance of the optimization problem with respect to the state. Thus, we can rewrite the problem 1.3 as

$$\begin{aligned} &\min_{S} J(S) \\ &s.t. \\ &S \in \mathcal{S} \end{aligned} \tag{1.5}$$

In addition we will assume without of loss of generality that the centralized system is controlled only by two agents. The distribution of the centralized problem between the agents is not easy. In first place, the centralized cost function $J(S)$ gives raise to two different cost functions, $J_1(S)$ and $J_2(S)$, which are the objective functions of the agents. In this thesis we will assume that $J(S)$ is simply the sum of the sum of the cost functions of the agents, that is, $J(S) = J_1(S) + J_2(S)$. Likewise, the set of centralized variables S has to be decomposed in three different sets: S_1, S_2 and S_{12}, where S_1 and S_2 stand respectively for the decoupled

variables of agents 1 and 2 and S_{12} represent the set of coupling variables, that is, those variables that directly affect both J_1 and J_2. Unless the set S_{12} is empty, it is clear that the agents have to communicate somehow in order to find a coordinated solution for the problem 1.5. Finally, the difficulties imposed by the constraints will be relaxed and it will be assumed that there is no coupling, that is, $\mathcal{S} = \mathcal{S}_1 \times \mathcal{S}_2 \times \mathcal{S}_{12}$.

Next, we introduce the main approaches that can be found in the literature. It is important to remark that the fact of presenting the algorithms in an unified manner allows only to capture the essence of each scheme but may miss some important details. Therefore we encourage the reader to see the references that are given for each scheme.

1.6.1 Decentralized schemes

This solution is based on the following assumption: the set of variables S can be decomposed in two different sets S_1 and S_2 so that the optimization problem 1.5 is trivially parallelizable as follows

$$\begin{array}{lcl} \min\limits_{S} J(S) & & \min\limits_{S_1} J_1(S_1) \;+\; \min\limits_{S_2} J_2(S_2) \\ s.t. & \equiv & s.t. \qquad\qquad\quad s.t. \\ S \in \mathcal{S} & & S_1 \in \mathcal{S}_1 \qquad\quad S_2 \in \mathcal{S}_2 \end{array} \tag{1.6}$$

The decentralization of the control problem is possible when the system is composed of subsystems whose dynamics and constraints are decoupled or the coupling is negligible. In this case it is not necessary to establish any mean of communication between the agents. Alternatively, some authors duplicate variables in order to separate the problem [2].

Decentralized MPC schemes can be easily found in the literature. For example, in [60], an MPC algorithm was proposed under the main assumptions that the system is nonlinear, discrete-time and no information is exchanged between local controllers. The stability of this class of systems, from an input-to-state stability point of view, was studied in [79]. In [2] the centralized MPC problem is decentralized considering only a one step horizon, which guarantees small deviations in the values of the variables the agent share. In addition, a sufficient criterion for analyzing a posteriori the asymptotic stability of the process model in closed-loop with the set of decentralized MPC controllers is given. This work is enhanced in [3] for the case of packet loss. Finally, in [45] a decentralized control architecture for nonlinear systems with continuous and asynchronous measurements was presented.

1.6.2 DMPC based on information broadcast

In this category we include those DMPC schemes in which the agents communicate with the goal of providing useful information for the decisions of the rest of their neighbors. Thus, no negotiation procedure takes place between the agents. Under this approach we can consider that the set of shared variables S_{12} is empty but the cost function of each agent depends on both S_1 and S_2. For this reason agent 1 transmits to agent 2 information about his variables (S_1^c) and agent 2 sends to agent 1 information about his variables (S_2^c). The nature of this information depend on the particular algorithm; it can be a prediction of the future value of some variables, a set of future possible values, etc. One way to address this class of problems is to follow a worst case approach, that is, each agent tries to optimize its outcome for the worst possible decision of its neighbor solving a minimax optimization problem

$$\begin{array}{lcll} \min\limits_{S} J(S) & & \min\limits_{S_1}\max\limits_{S_2} J_1(S_1,S_2) & + \quad \min\limits_{S_2}\max\limits_{S_1} J_2(S_1,S_2) \\ s.t. & \cong & s.t. & s.t. \\ S \in \mathcal{S} & & S_1 \in \mathcal{S}_1 & S_2 \in \mathcal{S}_2 \\ & & S_2 \in S_2^c & S_1 \in S_1^c \end{array} \tag{1.7}$$

In [33] a DMPC scheme for linear systems coupled only through the state is considered. In this scheme the agents exchange the predictions about their state at the end of each sample step. In [34] the DMPC controllers exchange bounds of their state trajectories and incorporate this information into their local problems. The main drawback of this approach is that each agent solves a local min-max problem similar to (1.7) on each iteration with respect to the worst-case disturbances, which is a very conservative solution. A similar approach is followed by [86] in which subsystems with independent dynamics but coupled constraints are considered. In this work each agent optimizes its local cost function knowing the predicted plans for the other subsystems using a robust MPC approach. An extension of this work [94] proposes the use of tubes for DMPC. In this scheme, the agents exchange the region of the state space in which their future state trajectories will lie along a given prediction horizon. The agents only have to communicate when these predicted regions change. Otherwise each subsystem can remain into its tube without communicating with the rest of the agents. In [15], MPC scheme based on contractive constraints is applied for the distributed control of a power system. In this case the agents exchange their state predictions once at the beginning of the each control cycle. The stability of the scheme is assured by mean of a contractive constraint imposed on the first state in the prediction horizon. In [38] decentralized MPC of dynamically decoupled systems where the cost function and constraints couple the dynamical behavior of the systems was studied. It is remarkable that although the agents do not communicate, they can access to the state measurements of their neighbors. In [24], the problem of distributed control of dynamically decoupled

nonlinear systems coupled by their cost function is considered. This method is extended to the case of dynamically coupled nonlinear systems in [22] and applied as a distributed control strategy for supply chains in [23]. In this implementation, the agents optimize locally their own policy, which is communicated to their neighbors. The stability is assured through a compatibility constraint: the agents commit themselves not to deviate too far in their state and input trajectories from what their neighbors believe they plan to do. Another interesting work is [46], which is an evolution of [45]. In this paper a distributed model predictive control method for the design of networked control systems based on Lyapunov-based model predictive control was presented. In both cases, each agent had access to the full system model.

Other algorithms in the literature are based on an iterative procedure of information broadcast. In each sample time the agents exchange information and solve their local problem shown in equation (1.7). For example, in [98] this procedure is presented as communication-based control. In [62] another iterative implementation of a similar DMPC scheme was applied together with a distributed Kalman filter to a quadruple tank system. Finally, in [43] the Shell benchmark is used to test a similar algorithm. Note that all these methods lead in general to Nash equilibria as long as the cost functions of the agents are selfish.

1.6.3 DMPC based on agent collaboration

In this category we include those DMPC schemes in which the agents exchange information trying to obtain a consensus on the values of the shared variables. It has to be remarked that this category includes algorithms very different in their nature. In particular, two different approaches can be found in the literature. The first one consists on the distribution of the centralized optimization problem between the agents. Methods such as primal or dual decomposition are based on this idea. An extensive review of this kind of algorithms can be found in [11]. The second approach distributes the problem formulation between agents and establishes ways of negotiation between them so that joint decisions can be taken.

Primal decomposition

Primal decomposition algorithms are based on the following idea:

1. Each agent solves its optimization problem assuming that the coupling variable is fixed, that is,

$$\begin{array}{ll} \phi_1(S_{12}) = \min\limits_{S_1} J_1(S_1, S_{12}) & \phi_2(S_{12}) = \min\limits_{S_2} J_2(S_2, S_{12}) \\ s.t. & s.t. \\ S_1 \in \mathcal{S}_1 & S_2 \in \mathcal{S}_2 \end{array} \tag{1.8}$$

2. The distributed problem is reduced to a master problem of the complicating variable S_{12}. From a centralized point of view the problem is

$$\begin{array}{l} \min\limits_{S_{12}} \phi_1(S_{12}) + \phi_2(S_{12}) \\ s.t. \\ S_{12} \in \mathcal{S}_{12} \end{array} \tag{1.9}$$

   which can be solved in a distributed fashion. To this end, agent i calculates g_i, which is the subgradient (subderivative) of $\phi_i(S_{12})$ with respect to the complicating variable, that is, $g_i \in \phi_i(S_{12})$.

3. The complicating variable is updated

$$S_{12} = S_{12} - \alpha(g_1 + g_2)$$

   where α is the step size.

4. Return to step 1 until the difference between the complicating variable between two consecutive steps is below a given threshold.

Dual decomposition

This approach consists of creating local versions of the complicating variable with additional consistency constraints that enforce them to have the same value. In the case of two agents, we can decompose the the complicating variable S_{12} in two local versions, S_{12}^1 and S_{12}^2, one for each agent. The resulting optimization problem, also called primal problem, is

$$\begin{array}{lcl} \min\limits_{S} J(S) & & \min\limits_{S_1, S_{12}^1} J_1(S_1, S_{12}^1) \quad + \quad \min\limits_{S_2, S_{12}^2} J_2(S_2, S_{12}^2) \\ s.t. & \cong & s.t. \qquad\qquad\qquad\quad s.t. \\ S \in \mathcal{S} & & S_1 \in \mathcal{S}_1 \qquad\qquad\quad S_2 \in \mathcal{S}_2 \\ & & S_{12}^1 = S_{12}^2 \qquad\qquad\quad S_{12}^1 = S_{12}^2 \end{array} \tag{1.10}$$

As it can be seen, the problem (1.10) changes the coupling from the optimization variables to the constraints. However, this difficulty can be solved with the aid of the Lagrange multipliers. The lagrangian or dual function of the centralized cost function is

$$L(S_1, S_{12}^1, S_2, S_{12}^2) = J_1(S_1, S_{12}^1) + J_2(S_2, S_{12}^2) + \lambda^T(S_{12}^1 - S_{12}^2), \tag{1.11}$$

which can be split in two different functions of the local variables of the agents. Dual decomposition algorithms can be summarized in these steps:

1. Each agent i finds the values of (S_i, S_{12}^i) that minimizes its dual function with a fixed value of the lagrange multiplier λ.

$$\begin{array}{ll} \phi_1(\lambda) = \min\limits_{S_1, S_{12}^1} J_1(S_1, S_{12}^1) + \lambda S_{12}^1 & \phi_2(\lambda) = \min\limits_{S_2, S_{12}^2} J_2(S_2, S_{12}^2) - \lambda S_{12}^2 \\ s.t. & s.t. \\ S_1 \in \mathcal{S}_1 & S_2 \in \mathcal{S}_2 \end{array} \tag{1.12}$$

2. The distributed problem is reduced to a problem of the complicating variable. The minimum of the primal problem (1.10) is attained at the maximum of the lagrangrian (1.11) with respect λ. The problem

$$\max_{\lambda} \phi_1(\lambda) + \phi_2(\lambda) \tag{1.13}$$

   can be solved in a distributed fashion.

3. It can be proved that when λ is maximum, the variables S_{12}^1 and S_{12}^2 have the same value. That means that the minimum of (1.10) will be attained when the gradient of λ is zero. In order to obtain the maximum on λ, we can use a distributed gradient search

$$\lambda = \lambda - \alpha(S_{12}^2 - S_{12}^1)$$

   where α is the step size.

4. Return to step 1 until the enough precision has been obtained, that is, when the update of the variable λ is below a given threshold.

Note that if the algorithm is stopped before the convergence, it is likely that the solution obtained is not feasible, that is, $S_{12}^1 \neq S_{12}^2$. Nevertheless, this can be solved in part by taking $S_{12} = (S_{12}^1 + S_{12}^2)/2$.

Dual decomposition has been used for DMPC in [82]. An augmented lagrangian formulation is proposed in [71] and applied to the control of irrigation canals in [70]. The problem of dual decomposition is, in general, the same that primal decomposition has: it requires a great number of iterations to obtain a solution.

Jacobi algorithm

This algorithm is an iterative method for the parallel optimization of nonlinear problems and its description can be found in [11] (see pages 219-223). This algorithm is the core idea of one of Venkat's feasible cooperation-based MPC [99, 93, 83], which is one of the most popular approaches to solve the DMPC problem. Cooperation-based DMPC assumes that the set of coupling variables S_{12} is empty. The coupling comes in the cost functions of both agents, which depend both on S_1 and S_2. At each sample time, the agents begin an iterative procedure in order to reach a joint solution which can be described as follows:

1. Step 1. Begin the iteration procedure with $p = 0$.
2. Step 2. At iteration p each agent solves a centralized optimization problem assuming that the neighboring variables have the value of the previous iteration:

$$\begin{array}{ll} S_1^p = arg\min_{S_1} J(S_1, S_2^p) & S_2^p = arg\min_{S_2} J(S_1^p, S_2) \\ s.t. & s.t. \\ S_1 \in \mathcal{S}_1 & S_2 \in \mathcal{S}_2 \\ S_2^p = S_2^{p-1} & S_1^p = S_1^{p-1} \end{array} \tag{1.14}$$

3. If the algorithm converges, that is, $S_i^p = S_i^{p+1}$, or a maximum number of iterations has been exceeded, then the iteration procedure stops. Otherwise, increase the iteration index p and go to step 2.

Under certain assumptions it can be proved that this procedure is convergent. The sequence of iterates converges to an optimal limit point which coincides with the centralized MPC solution. This solution also holds the condition of Pareto optimality.

Note that the algorithm is somehow similar to the primal decomposition one. The difference between them is in the amount of centralized information the agents have. In this case all the agents optimize the centralized cost function with respect their input variables instead of only their particular cost functions. This difference allows the agents not to have to exchange the subgradient of the complicating variables. The price to pay is the extra centralized information that all the agent in the system must have.

Descent direction algorithm

This algorithm is another iterative method and it is a distributed version of the method of the feasible directions given in [74]. Again it is assumed that the set of coupling variables

S_{12} is empty. The coupling comes in the cost functions of both agents, which depend both on S_1 and S_2. In addition, it is assumed that the local cost functions include the effect of each agent control actions in the other agent objectives. Thus, the local cost functions are cooperative in their nature. In each time sample the agents begin an iterative procedure in order to reach a joint solution which can be described as follows:

1. Step1. Begin the iteration procedure with $p = 0$.
2. Step 2. At iteration p, agent one solves the following optimization problem

$$\begin{aligned} & S_1^{p+1} = arg \min_{S_1} \nabla_{S_1} J_1(S_1^p, S_2^p)^T (S_1 - S_1^p) \\ & s.t. \\ & S_1 \in \mathcal{S}_1 \end{aligned} \tag{1.15}$$

   while agent two solves

$$\begin{aligned} & S_2^{p+1} = arg \min_{S_2} \nabla_{S_2} J_2(S_1^p, S_2)^T (S_2 - S_2^p) \\ & s.t. \\ & S_2 \in \mathcal{S}_2 \end{aligned} \tag{1.16}$$

3. If the algorithm converges, that is, $S_i^p = S_i^{p+1}$, or a maximum number of iterations has been exceeded, then the iteration procedure stops. Otherwise, increase the iteration index p and go to step 2.

As it can be seen, this algorithm follows a gradient search to find the solution for the problem. This method has been applied in [14] to an urban traffic network. A different gradient-based distributed dynamic optimization method is proposed in [89] and applied to an experimental four tanks plant in [5]. The method of [89] is based on the exchange of sensitivities. This information is used to modify the local cost function of each agent adding a linear term which partially allow to consider the other agents' objectives.

1.7 Objectives of the thesis

As it has been seen in the literature review, there is a good number of distributed MPC techniques. Some of them, provide the same performance as centralized MPC, at least theoretically [71, 99]. However, obtaining optimal results is not an easy task and requires in general an intensive use of the communication network. The main objective of this thesis is

to provide a good control performance while minimizing the number of communications between the agents and guaranteeing properties such as the stability of the closed-loop system. Likewise, we have tried to minimize the amount of centralized information the agents need.

Some sacrifices have to be done in order to reduce the communicational burden of the control schemes developed in this work. The first one is to reduce the number of possible actions that can be implemented by the agents. This reduction aims to limit the disturbances that the agents induce to their neighbors and simplifies the search of a joint solution for the control problem. It is important to stand out that the decisions are taken collectively. Collective decision making proves to be a good strategy even in scenarios in which players are selfish [8]. In addition, we provide design methods that guarantee the stability of the closed-loop system for the distributed control strategies that are proposed in this thesis. Moreover, we have developed algorithms to obtain a particular type of invariant set for distributed systems which can be used to analyze decentralized and distributed control schemes. A second goal of the thesis has been to transpose results of distributed control to distributed estimation, which is natural since distributed estimation is needed in distributed applications and because the estimation problem is the dual of the control problem. Finally, our last objective has been to develope techniques based on the dynamical switching of the links of the network, so that links which provide small gains in the control performance are disabled. Based on this work, we have also proposed a method to determine what agents and links are more important in a distributed system.

1.8 Thesis outline

The outline of the thesis is the following:

- **Chapter 2: Distributed Model Predictive Control Based on Game Theory for Two Agents**. The interaction between two agents is simplified using an strategic game in which the number of possible control actions is reduced to three for each agent. The three possible choices are to cooperate with the other agent, to behave selfishly or to implement a stabilizing option. Agents choose the best decision of the possible nine combinations trying to optimize the global cost. The stability of the closed-loop system is guaranteed by the controller design method that we propose in the chapter. This control scheme is applied to a supply chain simulation example and a benchmark of the project HD-MPC, the four tank plant. In addition, the robustness of the algorithm is tested against data losses in a stirred tank reactor simulation example.
- **Chapter 3: Distributed Model Predictive Control Based on Game Theory for N Agents**. The combinatorial explosion of the distributed problem hinder the

extension of the previous scheme to a general case. For this reason, a simplified scheme is developed in which each agent sends proposal for the control actions to his neighbors. Neighbors answer quantifying the positive or negative impact of the proposal. Only those proposals that reduce the global cost are accepted. In case that no proposal is accepted a stabilizing control action is applied. The stability of the closed-loop system is guaranteed by the controller design method that we propose in the chapter. Likewise, we define the concept of jointly invariant set and propose a method for its obtention. This control scheme is applied to a supply chain and an irrigation canal simulation examples.

- **Chapter 4: Distributed Receding Horizon Kalman Filter**. Given that estimation is the dual problem of control, it is natural to apply distributed control techniques to the estimation problem In this chapter the estimation problem is reduced to a dynamic programming problem which is distributed between the agents by means of dual decomposition.

- **Chapter 5: Applications of Cooperative Game Theory to the Control and Estimation of Distributed Systems**. Cooperative game theory provides mathematical tools very appropriated for the analysis of situations of conflict in which the agents may establish binding agreements. In this chapter of the thesis some results of coalitional game theory are transposed to distributed control. Based on these results, a control scheme to manage dynamically the links of a network is developed. In addition, a method to analyze the relative importance of links and agents in a distributed system is proposed.

- **Chapter 6: Conclusions**. The thesis ends with a chapter that analyzes the most relevant contributions and, additionally, points out future research lines in the field of distributed systems.

1.9 Contributions

As it has been seen in this section, this thesis can be located at the intersection between the fields of distributed control and game theory. In this subsection we will enumerate the major contributions made in this work and the results from the thesis that have been published or submitted to conferences and journals.

Given the difficulty of the distributed control problem, it was natural to face a simplified case in the first years of work of the thesis. For this reason, we focused initially in the case of two agents. As a result of this we developed a distributed control scheme with low communications step. This scheme was put to test with a MIMO system described by its

transfer function in [51] and applied to systems described in state space in [53, 52]. The proposed control scheme provides a good performance and guarantees the stability of the closed-loop system under certain assumptions. Besides these three conference papers, this work has been published in an international journal in [54] is compared to other distributed schemes with a real benchmark in a paper submitted to a journal [5].

Once the problem of two agents was solved, we extended our methodology to the general case of systems with N agents. In order to overcome the difficulties of the combinatorial explosion, a simplified version of the scheme was developed and as a result of this a negotiation algorithm was proposed. As our experiments show, the control scheme provides a good performance in a low number of communication steps. Again, a controller design method that guarantees the stability of the closed-loop system is provided. Additionally, a method to compute the jointly distributed invariant set of the system is presented. This work has been submitted for publication to an international journal [57, 102] and to an international conference [58].

Contributions have also been made to the distribution of the state estimation problem. Concretely, a technique based on the dual decomposition of the state estimation problem was developed together with professor Anders Rantzer, from the LTH in Lund (Sweden). This work has been accepted for publication in [50].

Besides these algorithms, other meaningful contributions to the state of the art have been made. In particular, it is remarkable the novel application of coalitional game theory to distributed control in order to dynamically decompose the agents in coalitions with low interaction. As a result of this we have proposed a distributed control scheme in which the decomposition of the centralized system is done dynamically. Additionally, the cooperative game theory tools provide an interesting method to analyze the relative relevance of agents and links in a network. This work has been presented in [48] and submitted to an international conference [55] and a journal [56].

Chapter 2

Distributed Model Predictive Control Based on Game Theory for Two Agents

As it was stated in the introduction, one of the goals of this thesis is to design distributed cooperative controllers with low communicational burden. In this chapter we focus on the particular case in which the system is divided in two subsystems. In particular, we propose a distributed model predictive control scheme based on a cooperative game in which two different agents communicate in order to find a solution to the problem of controlling two constrained linear systems coupled through the inputs. We assume that each agent only has partial information of the model and the state of the system. In the proposed scheme, the coordination problem between the agents is reduced to a team game where they have to choose one out of three options. To this end, the agents communicate twice each sampling time in order to share enough information to take a cooperative decision. Concretely, we provide sufficient conditions that guarantee practical stability of the closed-loop system as well as an optimization based procedure to design the controller so that these conditions are satisfied. In addition, we study the robustness of the distributed scheme against data losses due to failures in the communication channel. The theoretical results and the design procedure are illustrated using different simulation examples. Finally, we show the results that the control scheme has got in a benchmark of the european project "Hierarchical and distributed model predictive control" (HD-MPC)[1].

The outline of the chapter is as follows. In section 2.1 the proposed DMPC scheme for two agents is presented. Section 2.2 shows the stability properties of the proposed scheme.

[1]HD-MPC project, contract number INFSO-ICT-223854.

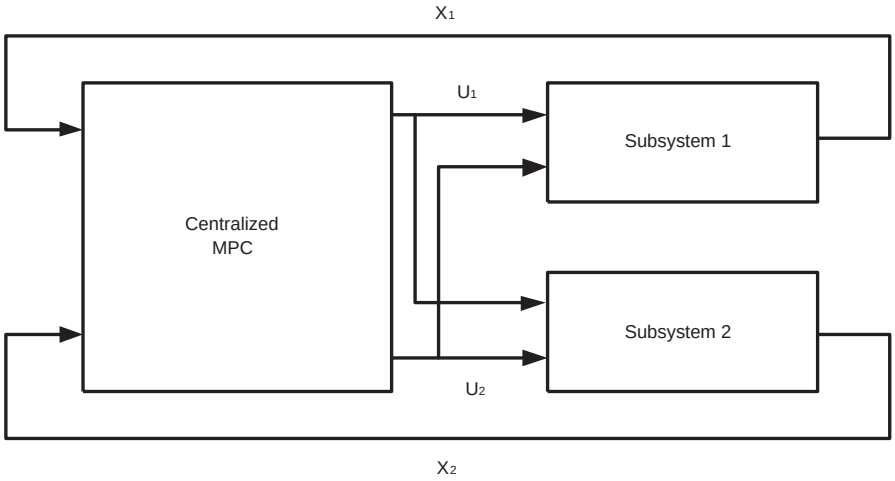


Figure 2.1: Centralized MPC scheme.

The next section shows different simulation examples to illustrate different features of the algorithm. A supply chain and two double integrators coupled through the inputs are used as examples. Section 2.4 tests the robustness of the scheme against communicational failures using a stirred tank reactor. In section 2.5 a real application of the control scheme to a four tank plant is shown. In addition, the controller is compared to other distributed algorithms in this section. Finally, conclusions are presented in section 2.6.

This chapter is based on the results and ideas published in [51, 53, 52, 54, 64].

2.1 Problem formulation

Consider the following class of distributed linear systems in which two subsystems coupled with the neighbor subsystem through the inputs are defined

$$\begin{aligned} x_1(t+1) &= A_1 x_1(t) + B_{11} u_1(t) + B_{12} u_2(t) \\ x_2(t+1) &= A_2 x_2(t) + B_{21} u_1(t) + B_{22} u_2(t) \end{aligned} \tag{2.1}$$

where $x_i \in \mathbb{R}^{n_i},\ i = 1, 2$ are the states of each subsystem and $u_i \in \mathbb{R}^{m_i},\ i = 1, 2$ are the different inputs. This class of systems are of relevance when identifications techniques are used to obtain the transfer function of a process. We consider the following linear constraints in the state and the inputs

$$x_i \in \mathcal{X}_i, u_i \in \mathcal{U}_i,\ i = 1, 2$$

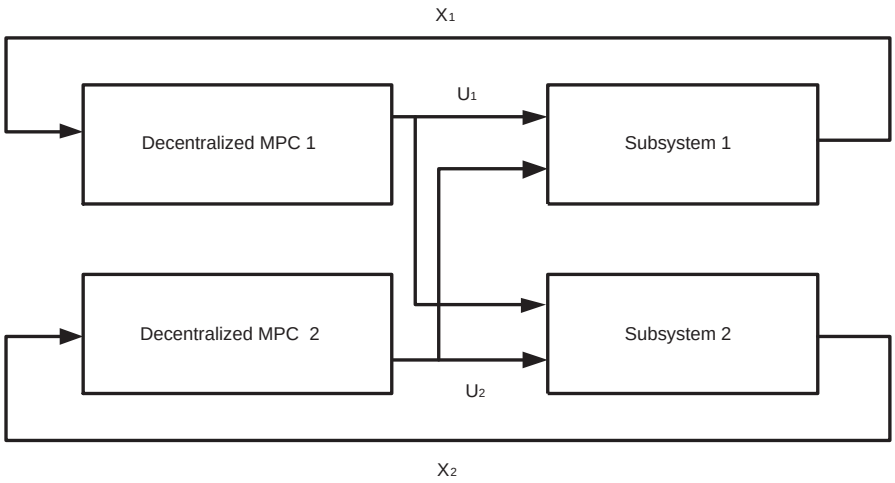


Figure 2.2: Decentralized MPC scheme.

where $\mathcal{X}_i$ and $\mathcal{U}_i$ with $i = 1, 2$ are defined by a set of linear inequalities.

The control objective is to regulate the system to the origin while guaranteeing that the constraints are satisfied. Centralized MPC solves a single optimization problem to decide the optimal sequences of the inputs u_1 and u_2 with respect to a given performance index based on the full model of the system and on measurements from all the sensors, see figure 2.1. In distributed and decentralized schemes two independent controllers (hereby denoted agents) are defined. Agent 1 has access to the model of subsystem 1, its state x_1 and decides the value of u_1. On the other hand, agent 2 has access to the model of subsystem 2, its state x_2 and decides the value of u_2. This implies that neither agent has access to the full model or state information and that in order to find a cooperative solution, they must communicate.

A control system is decentralized if there is no communication among the agents, see figure 2.2. This is the worst scenario from the performance point of view because each agent has to cope alone with its control problem with the risk that the absence of coordination in the agents decisions may lead to the instability of the system. The control system is distributed if there is communication between agents, see figure 2.3. The degree of communication depends on the control problem and the communication constraints. In this section we present a distributed MPC controller based on a cooperative game scheme between two different agents.

The objective of the proposed DMPC scheme is to minimize a performance index that

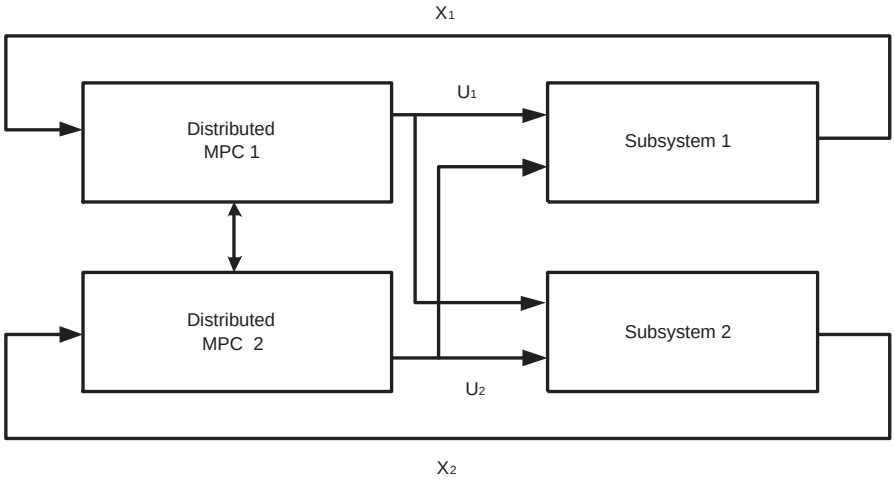


Figure 2.3: Distributed MPC scheme.

depends on the future evolution of both states and inputs. At each sampling time, each agent solves a sequence of reduced dimension optimization problems based on the model of its subsystem and assuming a given fixed input trajectory for its neighbor. In order to describe the algorithm, we need to introduce the following definitions:

- U_i: Future input sequence of agent i. These are the decision variables of the optimization problems solved by both agents.

$$U_1 = \begin{bmatrix} u_{1,0} \\ u_{1,1} \\ \vdots \\ u_{1,N-1} \end{bmatrix}, \; U_2 = \begin{bmatrix} u_{2,0} \\ u_{2,1} \\ \vdots \\ u_{2,N-1} \end{bmatrix}$$

- ni: Neighboring agent of agent i; that is, $U_{n1} = U_2$ and $U_{n2} = U_1$.

- J_i: Local cost function of agent i based on the predicted trajectories of its state defined as follows:

$$J_1(x_1, U_1, U_2) = \sum_{k=0}^{N-1} L_1(x_{1,k}, u_{1,k}) + F_1(x_{1,N})$$
$$J_2(x_2, U_2, U_1) = \sum_{k=0}^{N-1} L_2(x_{2,k}, u_{2,k}) + F_2(x_{2,N})$$

where $L_i(\cdot)$ and $F_i(\cdot)$ with $i = 1, 2$ are the stage and terminal cost functions respectively defined as:

$$L_i(x, u) = x^T Q_i x + u^T R_i u$$
$$F_i(x) = x^T P_i x$$

with $Q_i, P_i > 0, R_i \geq 0$. The prediction horizon is N. We use the notation $x_{i,k}$ to denote the k-steps ahead future predicted state obtained from the initial state x_i applying the input trajectories defined by and U_1 and U_2. Note that the second and third parameters of functions J_1 and J_2 are switched. This will allow us to simplify the algorithm definition.

- $U_i^d(t)$: Optimal input sequence of agent i at time t, denoted $U_i^d(t)$, defined as:

$$U_1^d(t) = \begin{bmatrix} u_{1,0}^d \\ u_{1,1}^d \\ \vdots \\ u_{1,N-1}^d \end{bmatrix}, \; U_2^d(t) = \begin{bmatrix} u_{2,0}^d \\ u_{2,1}^d \\ \vdots \\ u_{2,N-1}^d \end{bmatrix}$$

- $U_i^s(t)$: Shifted optimal input sequence of agent i obtained from the optimal input sequence of agent i at time $t-1$, denoted $U_i^d(t-1)$, as follows:

$$U_1^s(t) = \begin{bmatrix} u_{1,1}^d \\ u_{1,2}^d \\ \vdots \\ u_{1,N-1}^d \\ K_1 x_{1,N} \end{bmatrix}, \; U_2^s(t) = \begin{bmatrix} u_{2,1}^d \\ u_{2,2}^d \\ \vdots \\ u_{2,N-1}^d \\ K_2 x_{2,N} \end{bmatrix}$$

where $x_{1,N}, x_{2,N}$ are the N-steps ahead predicted state obtained from $x_1(t-1), x_2(t-1)$ respectively applying the input trajectories $U_1^d(t-1), U_2^d(t-1)$ and K_1, K_2 are two known feedback gains.

The proposed DMPC algorithm is the following:

1. At time step t, each agent i receives its corresponding partial state measurement $x_i(t)$.
2. Each agent i minimizes J_i assuming that the neighbor keeps applying the optimal trajectory evaluated at the previous time step; that is, $U_{ni} = U_{ni}^s(t)$. Agent 1 solves the following optimization problem:

$$\begin{aligned} U_1^*(t) = \arg\min_{U_1} \quad & J_1(x_1(t), U_1, U_2^s(t)) \\ & x_{1,k+1} = A_1 x_{1,k} + B_{11} u_{1,k} + B_{12} u_{2,k} \\ & x_{1,0} = x_1(t) \\ & x_{1,k} \in \mathcal{X}_1, \; k = 0, \ldots N \\ & u_{1,k} \in \mathcal{U}_1, \; k = 0, \ldots N-1 \\ & x_{1,N} \in \Omega_1 \end{aligned} \tag{2.2}$$

Agent 2 solves the following optimization problem:

$$\begin{aligned} U_2^*(t) = \arg\min_{U_2} \quad & J_2(x_2(t), U_2, U_1^s(t)) \\ & x_{2,k+1} = A_2 x_{2,k} + B_{22} u_{2,k} + B_{21} u_{1,k} \\ & x_{2,0} = x_2(t) \\ & x_{2,k} \in \mathcal{X}_2,\ k = 0, \dots N \\ & u_{2,k} \in \mathcal{U}_2,\ k = 0, \dots N-1 \\ & x_{2,N} \in \Omega_2 \end{aligned} \tag{2.3}$$

The sets Ω_1 and Ω_2 define the terminal region constraints that are necessary to prove closed-loop practical stability following a terminal region/terminal cost approach. Note that in both optimization problems the free variable is U_i while the neighbor input trajectory U_{ni} is fixed.

3. Each agent i minimizes J_i optimizing the neighbor input assuming that it applies the input trajectory computed in the previous optimization problem U_i^*. Agent 1 solves the following optimization problem:

$$\begin{aligned} U_2^w(t) = \arg\min_{U_2} \quad & J_1(x_1(t), U_1^*(t), U_2) \\ & x_{1,k+1} = A_1 x_{1,k} + B_{11} u_{1,k} + B_{12} u_{2,k} \\ & x_{1,0} = x_1(t) \\ & x_{1,k} \in \mathcal{X}_1,\ k = 0, \dots N \\ & u_{2,k} \in \mathcal{U}_2,\ k = 0, \dots N-1 \\ & x_{1,N} \in \Omega_1 \end{aligned} \tag{2.4}$$

Agent 2 solves the following optimization problem:

$$\begin{aligned} U_1^w(t) = \arg\min_{U_1} \quad & J_2(x_2(t), U_2^*(t), U_1) \\ & x_{2,k+1} = A_2 x_{2,k} + B_{22} u_{2,k} + B_{21} u_{1,k} \\ & x_{2,0} = x_2(t) \\ & x_{2,k} \in \mathcal{X}_2,\ k = 0, \dots N \\ & u_{1,k} \in \mathcal{U}_1,\ k = 0, \dots N-1 \\ & x_{2,N} \in \Omega_2 \end{aligned} \tag{2.5}$$

In this optimization problem the free variable is U_{ni} (the input trajectory U_i is fixed). Solving this optimization problem, agent i defines an input trajectory for its neighbor that optimizes its local cost function J_i.

4. Both agents communicate. Agent 1 sends $U_1^*(t)$ and $U_2^w(t)$ to agent 2 and receives $U_2^*(t)$ and $U_1^w(t)$.

5. Each agent evaluates the local cost function J_i for each the nine different possible combination of input trajectories; that is $U_1 \in \{U_1^s(t), U_1^w(t), U_1^*(t)\}$ and $U_2 \in \{U_2^s(t), U_2^w(t), U_2^*(t)\}$.

Table 2.1: Cost function table used for the decision making.

	$U_2^s(t)$	$U_2^*(t)$	$U_2^w(t)$
$U_1^s(t)$	$J_1(x_1(t), U_1^s(t), U_2^s(t))$ $+J_2(x_2(t), U_2^s(t), U_1^s(t))$	$J_1(x_1(t), U_1^s(t), U_2^*(t))$ $+J_2(x_2(t), U_2^*(t), U_1^s(t))$	$J_1(x_1(t), U_1^s(t), U_2^w(t))$ $+J_2(x_2(t), U_2^w(t), U_1^s(t))$
$U_1^*(t)$	$J_1(x_1(t), U_1^*(t), U_2^s(t))$ $+J_2(x_2(t), U_2^s(t), U_1^*(t))$	$J_1(x_1(t), U_1^*(t), U_2^*(t))$ $+J_2(x_2(t), U_2^*(t), U_1^*(t))$	$J_1(x_1(t), U_1^*(t), U_2^w(t))$ $+J_2(x_2(t), U_2^w(t), U_1^*(t))$
$U_1^w(t)$	$J_1(x_1(t), U_1^w(t), U_2^s(t))$ $+J_2(x_2(t), U_2^s(t), U_1^w(t))$	$J_1(x_1(t), U_1^w(t), U_2^*(t))$ $+J_2(x_2(t), U_2^*(t), U_1^w(t))$	$J_1(x_1(t), U_1^w(t), U_2^w(t))$ $+J_2(x_2(t), U_2^w(t), U_1^w(t))$

6. Both agents communicate and share the information of the value of local cost function for each possible combination of input trajectories. In this step, both agents receive enough information to take a cooperative decision.

7. Each agent applies the input trajectory that minimizes $J = J_1 + J_2$. Because both agents have access to the same information after the second communication cycle, both agents choose the same optimal input sets. We denote the chosen set of input trajectories as $U_1^d(t), U_2^d(t)$.

8. The first input of each optimal sequence is applied and the procedure is repeated the next sampling time.

From a game theory point of view, at each time step both agents are playing a cooperative game. This game can be synthesized in strategic form by a three by three matrix. Each row represents one of the three possible decisions of agent 1, and each column represents one of the three possible decisions of agent 2. The cells contain the sum of the cost functions of both agents for a particular choice of future inputs. At each time step, the option that yields a lower global cost is chosen. Note that both agents share this information, so they both choose the same option. The nine possibilities are shown in table 2.1.

Remark. At each sampling time, the controllers decide among three different options. The shifted optimal input trajectory $U_i^s(t)$ keeps applying the latest optimal trajectory. The *selfish* option $U_i^*(t)$ provides the best improvement in J_i if the rest of the systems manipulated variables stay unchanged. The *altruist* option $U_i^w(t)$ provides the best improvement for the neighbor agent cost function J_2. In this case, the agent i sacrifices its own welfare in order to improve the overall performance.

Remark. Centralized MPC solves a single large-scale problem based on the model of the whole system, see figure 2.1. In the example section we will compare the performance of the proposed approach with a centralized MPC controller based on the following optimization

problem:

$$\begin{aligned}\{U_1^c(t), U_2^c(t)\} = \arg\min_{U_1,U_2} \quad & J_1(x_1(t), U_1, U_2) + J_2(x_1(t), U_2, U_1) \\ & x_{1,k+1} = A_1 x_{1,k} + B_{11} u_{1,k} + B_{12} u_{2,k} \\ & x_{1,0} = x_1(t) \\ & x_{1,k} \in \mathcal{X}_1,\ k = 0, \dots N \\ & u_{1,k} \in \mathcal{U}_1,\ k = 0, \dots N-1 \\ & x_{1,N} \in \Omega_1 \\ & x_{2,k+1} = A_2 x_{2,k} + B_{22} u_{2,k} + B_{21} u_{1,k} \\ & x_{2,0} = x_2(t) \\ & x_{2,k} \in \mathcal{X}_2,\ k = 0, \dots N \\ & u_{2,k} \in \mathcal{U}_2,\ k = 0, \dots N-1 \\ & x_{2,N} \in \Omega_2\end{aligned} \tag{2.6}$$

The centralized MPC provides in general the best closed-loop performance, but can only be applied when it is possible to control the system with a single controller that has access to the full model and state of the same.

Remark. In general, the minimum number of communication steps needed for a cooperating control scheme is two. In the first step each agent informs of its intentions to its neighbors and during the second it can confirm if it accepts its neighbors' intentions. In the best case an agreement can be achieved in the second step, but in general an iterative procedure will be needed to reach an agreement.

Remark. The proposed controller scheme is cooperative from a game theory point of view because each agent chooses the solution that optimizes a cost function that depends on both subsystems, not only on the future trajectories of its subsystem. If the decision taken does not depend on a global performance index, the solution is not cooperative. In the simulation section we will compare the proposed distributed controller with a distributed scheme in which the two agents communicate, but do not take a cooperative decision. They iterate until an agreement is obtained. In this case, the solution is a Nash equilibrium [98] of the multi-objective optimization problem defined by the cost functions of both agents. At each iteration, agent 1 solves the following optimization problem:

$$\begin{aligned}U_1^{l+1} = \arg\min_{U_1} \quad & J_1(x_1, U_1, U_2^l) \\ & x_{1,k+1} = A_1 x_{1,k} + B_{11} u_{1,k} + B_{12} u_{2,k} \\ & x_{1,0} = x_1 \\ & x_{1,k} \in \mathcal{X}_1,\ k = 0, \dots N \\ & u_{1,k} \in \mathcal{U}_1,\ k = 0, \dots N-1 \\ & x_{1,N} \in \Omega_1\end{aligned} \tag{2.7}$$

and agent 2 solves the following optimization problem:

$$
\begin{aligned}
U_2^{l+1} = \arg\min_{U_2} \quad & J_2(x_2, U_2, U_1^l) \\
& x_{2,k+1} = A_2 x_{2,k} + B_{22} u_{2,k} + B_{21} u_{1,k} \\
& x_{2,0} = x_2 \\
& x_{2,k} \in \mathcal{X}_2,\ k = 0, \dots N \\
& u_{2,k} \in \mathcal{U}_2,\ k = 0, \dots N-1 \\
& x_{2,N} \in \Omega_2
\end{aligned}
\tag{2.8}
$$

with $U_1^0 = U_1^s$ and $U_2^0 = U_2^s$; that is, the initial guess is given by the shifted trajectory. An agreement is reached when the difference between the proposed control vector by each agent at one iteration and its value at the previous iteration is below a threshold. This implies, that they do not share information about the utility of each decision, they reach an agreement when neither of them can improve, hence reaching a Nash equilibrium. In the example we will compare the proposed controller with different controllers based on this distributed scheme, each one carrying out a fixed number of iterations, to demonstrate that the proposed cooperative scheme provides a better performance with a lower number of iterations.

Remark. Although the option chosen by the algorithm is the Pareto optimum of the game that both agents are playing, in general it is not a Pareto optimum of the multi-objective optimization problem defined by the cost functions J_1 and J_2.

Remark. The proposed scheme can be extended to deal with N agents, however, in order to build a global cost table to take a cooperative decision, the complexity grows exponentially. In order to reduce the complexity, the structure of the system may be exploited taking into account that an input may not affect all the outputs. Also, in general not all the possible cooperation options are employed with the same frequency, so it is possible to reduce further the complexity by not taking into account the less frequent options. In the next chapter, we propose a distributed scheme for the case of N agents following a slightly different approach: each agent optimizes with respect to all the manipulated variables that affect its dynamics. After that, the agent may make different proposals for the value of the set (or subsets) of these variables. In this way, the combinatorial explosion of the general case is avoided.

Remark. The computational burden of the proposed distributed scheme is in general lower than the one corresponding to the centralized scheme not only because the optimization problems are of a lower dimension (smaller number of free variables), but also because the agents can operate in parallel.

Remark. In the proposed algorithm both agents can operate in parallel; that is, the agents can compute U_i^* and U_i^w simultaneously (steps 2 and 3).

2.2 Stability properties

Controlling a system between two independent agents may lead to an unstable system. The resulting closed-loop system is a multiprocess system and studying the stability of this class of systems is in general a difficult task. Following a terminal region/terminal constraint approach [61, 26], in this section we provide sufficient conditions that guarantee practical stability of the closed-loop system as well as an optimization based procedure to design the controller so that these conditions are satisfied. This result is stated in the following theorem:

Theorem 1 *Assume that there exist linear feedbacks* $u_1 = K_1 x_1$ *and* $u_2 = K_2 x_2$ *such that the following conditions hold:*

$$F_1((A_1 + B_{11}K_1)x_1 + B_{12}K_2x_2) - F_1(x_1) + L_1(x_1, K_1x_1) - d_1 \leq 0,\ \forall x_2 \in \Omega_2 \quad (2.9a)$$

$$F_2((A_2 + B_{22}K_2)x_2 + B_{21}K_1x_1) - F_1(x_2) + L_1(x_2, K_2x_2) - d_2 \leq 0,\ \forall x_1 \in \Omega_1 \quad (2.9b)$$

$$x_1 \in \Omega_1 \rightarrow (A_1 + B_{11}K_1)x_1 + B_{12}K_2x_2 \in \Omega_1,\ \forall x_2 \in \Omega_2 \quad (2.9c)$$

$$x_2 \in \Omega_2 \rightarrow (A_2 + B_{22}K_2)x_2 + B_{21}K_1x_1 \in \Omega_2,\ \forall x_1 \in \Omega_1 \quad (2.9d)$$

$$K_1x_1 \in \mathcal{U}_1,\ \forall x_1 \in \Omega_1 \quad (2.9e)$$

$$K_2x_2 \in \mathcal{U}_2,\ \forall x_2 \in \Omega_2 \quad (2.9f)$$

$$\Omega_1 \in \mathcal{X}_1 \quad (2.9g)$$

$$\Omega_2 \in \mathcal{X}_2 \quad (2.9h)$$

Then, if at $t = 0$, $U_1^s(0), U_2^s(0)$ *are given such that Problems (2.2) and (2.3) are feasible for* $x_{1,0} = x_1(0), x_{2,0} = x_2(0), U_1 = U_1^s(0)$ *and* $U_2 = U_2^s(0)$, *then the proposed algorithm is feasible for all time steps* $t \geq 0$ *and system* (2.1) *in closed-loop with the proposed distributed MPC controller is ultimately bounded in a region that contains the origin in its interior.*

Proof:

The proof consists of two parts. We first prove recursive feasibility of Problems (2.2) and (2.3) if at time t, $U_1^s(t), U_2^s(t)$ are given such that (2.2) and (2.3) are feasible for $x_{1,0} = x_1(t), x_{2,0} = x_2(t), U_1 = U_1^s(t)$ and $U_2 = U_2^s(t)$. Then we prove that, under the stated assumptions,

$$J(t) = J_1(x_1(t), U_1^d(t), U_2^d(t)) + J_2(x_2(t), U_2^d(t), U_1^d(t))$$

is a decreasing sequence of values with a lower bound. This implies that system (2.1) in closed-loop with the proposed distributed MPC controller is ultimately bounded in a region that contains the origin in its interior.

Part 1. We will prove this part by recursion. First, we prove that if the state and input trajectories obtained from $x_1(t-1)$ and $x_2(t-1)$ applying $U_1^d(t-1)$ and $U_2^d(t-1)$ satisfy all the constraints of Problems (2.2) and (2.3), then $U_1^d(t)$ and $U_2^d(t)$ also satisfy all the constraints. Recalling step 5 of the proposed algorithm, to prove this statement it is sufficient to prove that there exists at least a pair of input trajectories that satisfy all the constraints. To this end, we will prove that $U_1^s(t), U_2^s(t)$ provide a feasible solution for $x_1(t)$ and $x_2(t)$. Note that in general, it is not possible to guarantee that any of the other options are feasible.

Taking into account that by definition $U_1^d(t-1)$ and $U_1^d(t-1)$ satisfy the constraints of Problems (2.2) and (2.3), the following statements hold

$$\begin{aligned}
&x_{1,k} \in \mathcal{X}_1,\ k = 0, \dots N \\
&u_{1,k}^d \in \mathcal{U}_1,\ k = 0, \dots N-1 \\
&x_{1,N} \in \Omega_1 \\
&x_{2,k} \in \mathcal{X}_2,\ k = 0, \dots N \\
&u_{2,k}^d \in \mathcal{U}_2,\ k = 0, \dots N-1 \\
&x_{2,N} \in \Omega_2
\end{aligned}$$

where $x_{1,k}, x_{2,k}$ are the k-steps ahead predicted state obtained from $x_1(t-1), x_2(t-1)$ respectively applying the input trajectories $U_1^d(t-1), U_2^d(t-1)$ defined by $u_{1,k}^d, u_{2,k}^d$ with $k = 0, \dots, N-1$.

At time step $t-1$, the first input of the chosen trajectories $U_1^d(t-1), U_2^d(t-1)$ are applied; that is, $u_1(t-1) = u_{1,0}^d$ and $u_2(t-1) = u_{2,0}^d$. This implies that

$$x_1(t) = A_1 x_1(t-1) + B_{11} u_1(t-1) + B_{12} u_2(t-1) = A_1 x_{1,0} + B_{11} u_{1,0}^d + B_{12} u_{2,0}^d = x_{1,1}.$$

Taking into account the definitions of $U_1^s(t)$ and $U_2^s(t)$, it can be proved that the k-steps ahead predicted state obtained from $x_1(t), x_2(t)$ respectively applying the input trajectories $U_1^s(t), U_2^s(t)$ satisfy all the constraints from $k = 0$ to $N-1$. Moreover, as

$$x_{1,N} \in \Omega_1,\ x_{2,N} \in \Omega_2$$

it holds that

$$\begin{aligned}
&(A_1 + B_{11}K_1)x_{1,N} + B_{12}K_2 x_{2,N} \in \Omega_1 \\
&(A_2 + B_{22}K_2)x_{2,N} + B_{21}K_1 x_{1,N} \in \Omega_2
\end{aligned}$$

and hence all the constraints of Problems (2.2) and (2.3) are satisfied which implies that $U_1^s(t), U_2^s(t)$ and hence $U_1^d(t), U_2^d(t)$ provide a feasible solution for $x_1(t)$ and $x_2(t)$. Taking into account that by assumption, $U_1^s(0), U_2^s(0)$ satisfy all the constraints for $x_1(0)$ and $x_2(0)$ and using the above result recursively, the statement of this part is proved.

Part 2. In this part we will prove that

$$J_1(x_1(t), U_1^s(t), U_2^s(t)) + J_2(x_2(t), U_2^s(t), U_1^s(t)) \leq J(t-1) + d_1 + d_2$$

where

$$J(t-1) = J_1(x_1(t-1), U_1^d(t-1), U_2^d(t-1)) + J_2(x_2(t-1), U_2^d(t-1), U_1^d(t-1)).$$

Taking into account the definitions of $U_1^d(t-1)$ and $U_1^s(t)$ it follows that

$$J_1(x_1(t), U_1^s(t), U_2^s(t)) - J_1(x_1(t-1), U_1^d(t-1), U_2^d(t-1))$$

is equal to

$$F_1((A_1 + B_{11}K_1)x_{1,N} + B_{12}K_2x_{2,N}) - F_1(x_{1,N}) + L_1(x_{1,N}, K_1x_{1,N}) - L_1(x_{1,0}, K_1x_{1,0})$$

As $L_1(x_{1,0}, K_1x_{1,0}) \geq 0$ and taking into account (2.9a) and (2.9b), that $x_{1,N} \in \Omega_1$ and that $x_{2,N} \in \Omega_2$ it follows that

$$J_1(x_1(t), U_1^s(t), U_2^s(t)) - J_1(x_1(t-1), U_1^d(t-1), U_2^d(t-1)) - d_1 \leq 0$$

Following the same steps for J_2 we obtain that

$$J_2(x_2(t), U_2^s(t), U_1^s(t)) - J_2(x_2(t-1), U_2^d(t-1), U_1^d(t-1)) - d_2 \leq 0$$

and hence

$$J_1(x_1(t), U_1^s(t), U_2^s(t)) + J_2(x_2(t), U_2^s(t), U_1^s(t)) \leq J(t-1) + d_1 + d_2$$

As the proposed algorithm chooses $U_1^d(t), U_2^d(t)$ as the pair of input trajectories that yield the minimum cost, it is easy to see that

$$J(t) \leq J(t-1) + d_1 + d_2$$

Following standard Lyapunov arguments and taking into account that recursive feasibility is guaranteed (see the first part of the proof), it is proved that system (2.1) in closed-loop with the MPC controller defined by the proposed controller is ultimately bounded in a region that contains the origin in its interior.

■

Remark. Theorem 1 guarantees that the closed-loop system is ultimately bounded in a closed region that contains the origin. However, it is possible to prove that the proposed controller provides asymptotic stability if the assumptions 2.9a and 2.9b are modified so that asymptotic stability of the centralized system is guaranteed. In the next chapter, we will study this topic in depth.

2.2.1 Design procedure

In the previous section, we have provided sufficient conditions to guarantee that the closed-loop system with the proposed distributed MPC scheme is practically stable. In general, designing the controller parameters so that these conditions are satisfied is a hard problem because the design constraints are coupled; for example, the constraints that define the invariant sets Ω_1 depend on the set of Ω_2 and viceversa. For centralized MPC controllers, there are various methods described in the literature on how to design a stabilizing local controller, terminal cost function and terminal region [61, 26, 37] (for example, the local controller and the terminal cost can be obtained solving a LQR problem). These results however cannot be applied to the distributed case. In this section we present an optimization based procedure to find local controllers K_1, K_2, matrices P_1, P_2 and regions Ω_1, Ω_2 such that (2.9) holds for a given system.

The procedure determines first matrices K_1, K_2, P_1 and P_2 such that (2.9a) and (2.9b) hold for any given sets Ω_1 and Ω_2 solving a linear matrix inequality (LMI) optimization problem. Once the local feedbacks K_1 and K_2 are fixed, the invariant sets Ω_1 and Ω_2 are obtained. Note that constants d_1 and d_2 are determined a posteriori, once the local feedbacks, terminal costs and terminal regions are fixed.

From the point of view of each agent, its neighbor's input can be viewed as a disturbance. This allows us to use well known tools from control of linear uncertain systems in order to determine a local controller such that a given degree of robustness is guaranteed. In [59, 1, 41] several methods to solve this class of problems are presented. In particular, constraint (2.9a) can be transformed into an LMI and solved using standard techniques, moreover, is equivalent to designing an $\mathcal{H}$-infinity controller for subsystem 1 assuming that u_2 is the disturbance [41]. The same technique can be followed to design K_2 and P_2. The following theorem defines an LMI constraint that only depends on the system matrices that guarantees that there exist K_1, K_2, P_1 and P_2 such that (2.9a) and (2.9b) hold.

Theorem 2 *Consider system (2.1). If there exist matrices W_i, Y_i and a constant γ_i such that the following inequality holds*

$$\begin{bmatrix} \gamma_i I & 0 & B_{i,ni}^T & 0 & 0 \\ * & W_i & W_i A_{ii}^T + Y_i^T B_{ii}^T & W_i Q_i^{\frac{1}{2}} & Y_i^T R_i^{\frac{1}{2}} \\ * & * & W_i & 0 & 0 \\ * & * & * & I & 0 \\ * & * & * & * & I \end{bmatrix} > 0. \tag{2.10}$$

then (2.9a) (or (2.9b), depending on the agent) is satisfied for $P_i = W_i^{-1}$, $K_i = Y_i W_i^{-1}$, and

$$d_i = \gamma_i \max_{x \in \Omega_{ni}} (K_{ni} x)^T K_{ni} x$$

Proof: In [41] it is proved that if (2.10) holds, then the following constraint is satisfied

$$F_i((A_i + B_{ii}K_i)x_i + B_{i,ni}v) - F_i(x_i) + L_i(x_i, K_i x_i) - \gamma_i v^T v \leq 0, \ \forall v \tag{2.11}$$

It follows that (2.9a) (or (2.9b), depending on the agent) holds for

$$d_i = \gamma_i \max_{x \in \Omega_{ni}} (K_{ni}x)^T K_{ni}x$$

■

Once the local controllers and the terminal cost functions are fixed, in order to design a distributed MPC scheme that satisfies the assumptions of Theorem 1 one needs to find sets Ω_1, Ω_2 such that (2.9c) to (2.9g) hold. In general this is a difficult problem because each of the sets depends on the other. The size of the terminal region for agent 1 is determined by the magnitude of the disturbances induced by its neighbor agent 2 and viceversa. We provide next an optimization based procedure to solve this problem. In order to present the algorithm we need the following definitions

Definition 1 *Given the following discrete-time linear system subject to bounded additive uncertainties*

$$x^+ = Ax + Bu + Dw$$

with $w \in \mathcal{W}$, subject to constraints in the state and the input $x \in \mathcal{X}, u \in \mathcal{U}$ and a linear feedback $u = Kx$; a set Ω is said to be a robust positive invariant set for the system if the following constraints hold

$$\begin{aligned} &x \in \Omega \rightarrow (A + BK)x + BKx \in \Omega, \ \forall w \in \mathcal{W} \\ &Kx \in \mathcal{U} \\ &\Omega \in \mathcal{X} \end{aligned}$$

Given system matrices A, B, D, K and the sets $\mathcal{X}$,$\mathcal{U}$,$\mathcal{W}$, there exists several methods to find a set Ω that satisfies these constraints, see for example [39] for a procedure to find the maximal robust positive invariant and [80] for a procedure to find an approximation of the minimal robust positive invariant. We denote $\Omega(A, B, D, K, \mathcal{X}, \mathcal{U}, \mathcal{W})$ the corresponding maximal robust positive invariant set.

Taking into account that the input of the neighbor agent can be considered as an unknown bounded disturbance, in order to decouple the computation of the sets Ω_1 and Ω_2, we use the following result based on finding a robust positive invariant set for each subsystem:

Theorem 3 *Given constants* $\lambda_1 \in (0,1]$ *and* $\lambda_2 \in (0,1]$*, if the sets defined as*

$$\Omega_1 = \Omega(A_1, B_{11}, B_{12}, \mathcal{X}_1, K_1, \lambda_1\mathcal{U}_1, \lambda_2\mathcal{U}_2)$$
$$\Omega_2 = \Omega(A_2, B_{22}, B_{21}, \mathcal{X}_2, K_2, \lambda_2\mathcal{U}_2, \lambda_1\mathcal{U}_1)$$

are not empty, then constraints (2.9c) to (2.9g) are satisfied.

Proof: The theorem is proved taking into account the definition of the operator Ω and that $\lambda_1\mathcal{U}_1 \subseteq \mathcal{U}_1$ and $\lambda_2\mathcal{U}_2 \subseteq \mathcal{U}_2$.

■

The main idea is that to determine the invariant sets both agents limit its inputs by a factor $\lambda_i \in (0,1]$ with $i = 1,2$ so the other agent can find the maximal robust positive invariant set with respect to a known bounded disturbance. For example, agent 1 finds the maximal robust positive invariant with respect to a disturbance bounded in $\lambda_2\mathcal{U}_2$ assuming that its input is bounded in $\lambda_1\mathcal{U}_1$. Agent 2 does the same. Note that these sets may be empty depending on the value of λ_1 and λ_2. If both sets exists, then they satisfy the stability constraints. In general an infinite number of possible values of λ_1 and λ_2 such that both sets are non empty may exist. In order to choose one, we propose to solve the following optimization problem to maximize the feasibility region of the distributed MPC controller:

$$\begin{aligned} \max_{\lambda_1\in(0,1],\lambda_2\in(0,1]} \quad & f(\Omega_1 \times \Omega_2) \\ & \Omega_1 = \Omega(A_1, B_{11}, B_{12}, K_1, \mathcal{X}_1, \lambda_1\mathcal{U}_1, \lambda_2\mathcal{U}_2) \\ & \Omega_2 = \Omega(A_2, B_{22}, B_{21}, K_2, \mathcal{X}_2, \lambda_2\mathcal{U}_2, \lambda_1\mathcal{U}_1) \end{aligned} \tag{2.12}$$

where $f(\cdot)$ is a measure of the size of a polyhedra (for example, its Chebyshev radius).

Once matrices K_1, K_2, P_1, P_2 and the sets Ω_1 and Ω_2 are determined, constants d_1 and d_2 can be calculated in order to obtain an estimation of the set in which the closed-loop system is ultimately bounded.

2.3 Simulation examples

In this section the theoretical results and the design procedure are illustrated using two different examples. The first example is focused on the controller design procedure. The second controller shows the application of the proposed approach to a supply chain problem. The simulations presented in this chapter were performed using Matlab in a computer equipped with a 2.2GHz Core 2 duo processor and 3 GB of RAM memory.

2.3.1 Two double integrators with coupled inputs

The system considered is composed by two double integrators with coupled inputs. The first subsystem is defined by the following matrices

$$A_{11} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix},\ B_{11} = \begin{bmatrix} 0 \\ 1 \end{bmatrix},\ B_{12} = \begin{bmatrix} 0 \\ 0.4 \end{bmatrix}$$

and the second subsystems is defined by

$$A_{22} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix},\ B_{22} = \begin{bmatrix} 0 \\ 1 \end{bmatrix},\ B_{21} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

The state and the input must satisfy the following constraints:

$$\|x_1\|_\infty \leq 1,\ \|x_2\|_\infty \leq 2,\ |u_1| \leq 1,\ |u_2| \leq 1$$

The stage cost functions of each agent are defined by $Q_i = I$ and $R_i = 1$ for $i = 1, 2$.

In order to determine the local controllers K_i and the corresponding weight matrices P_i that define the terminal cost function, a LMI problem based on (2.10) that minimizes the constant γ_i is solved for each agent. The following matrices are obtained:

$$K_i = \begin{bmatrix} -0.2023 & -0.9254 \end{bmatrix},\ i = 1, 2$$

$$P_1 = \begin{bmatrix} 32.6719 & -17.5149 \\ -17.5149 & 54.6366 \end{bmatrix},\ P_2 = \begin{bmatrix} 38.4509 & -5.6447 \\ -5.6447 & 50.1686 \end{bmatrix}$$

The last step necessary to apply the proposed algorithm is to determine an invariant region for the two agents, Ω_1 and Ω_2. Different approaches can be used to determine the values of λ_i that maximize the size of the terminal regions. In this example the terminal region was calculated for a grid with different values of λ_i. The criterion to select the maximum invariant region was the Chebyshev radius of the maximum ball inside the region. The results were $\lambda_1^* = 0.3$ and $\lambda_2^* = 0.5$. Figure 2.4 shows a 3D plot of the Chebyshev radius as a function of λ_1 and λ_2.

The constants λ_1 and λ_2 define a trade-off between the degree of freedom that the agents have in order to stabilize the system, and the size of the terminal region which determines the size of the disturbance. As λ_2 increases, the set defined by $K_2 x \in \lambda_2 \mathcal{U}_2$ increases. This implies that the set $\Omega_2 = \Omega(A_2, B_{22}, B_{21}, K_2, \mathcal{X}_2, \lambda_2\mathcal{U}_2, \lambda_1\mathcal{U}_1)$ becomes larger because the feasibility region of the input is larger, while the set $\Omega_1 = \Omega(A_1, B_{11}, B_{12}, K_1, \mathcal{X}_1, \lambda_1\mathcal{U}_1, \lambda_2\mathcal{U}_2)$ has to take into account bigger disturbances and may even cease to be defined (i.e., is empty). This

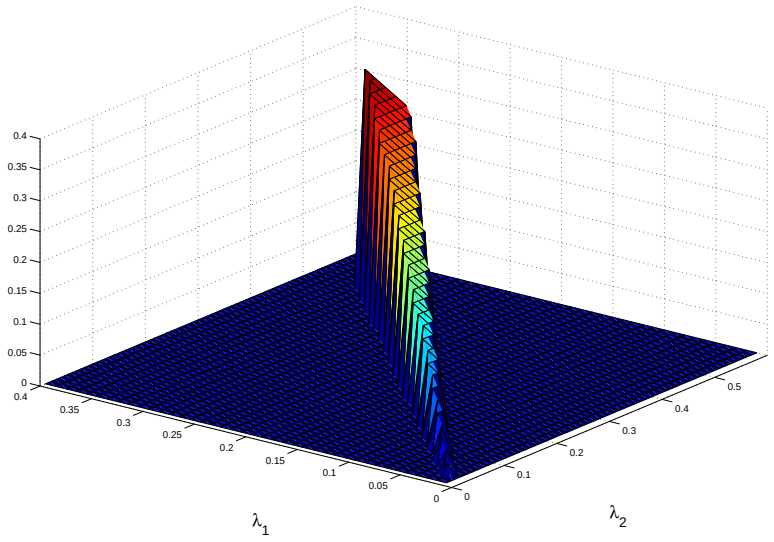


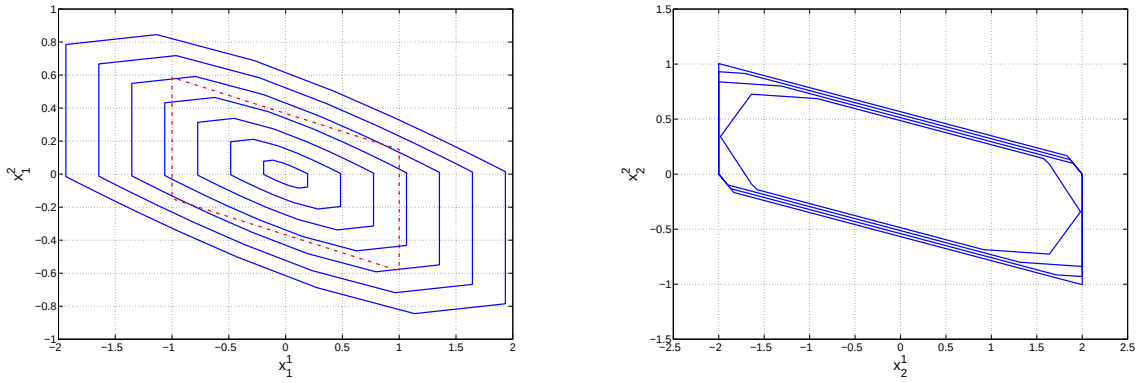
Figure 2.4: Chebyshev radius of the set $\Omega_1 \times \Omega_2$ for different values of λ_1 and λ_2.

happens when the minimum positive robust invariant set for an uncertainty bounded in $\lambda_2\mathcal{U}_2$ is not included in the feasibility region defined by $\mathcal{X}_1$ and $K_1x_1 \in \lambda_1\mathcal{U}_1$. In figure 2.5(a), inner approximations of the minimum positive invariant sets of subsystem 1 for different values of λ_2 and a fixed value of λ_1 are shown. It can be seen that for large values of λ_2, the inner approximation is not contained in the feasibility region of agent 2 (shown in red dashed line), and hence, it is empty. In figure 2.5(b) the maximum positive invariant set for the same values of λ_1 and a fixed value of λ_2 are shown. It can be seen how the size of the set always increases with λ_2.

In the first time step, a feasible solution for the centralized problem is used as the shifted trajectories. Simulations results are shown in the next figures for the following initial conditions:

$$x_1(0) = \begin{bmatrix} 0.7 \\ -1 \end{bmatrix}, \; x_2(0) = \begin{bmatrix} 1 \\ 0.8 \end{bmatrix}$$

Figure 2.6 shows the trajectories of the states of each agent, the inputs and the cost index. Figures 2.7(b) and 2.7(a) show the state trajectories of each agent along with its corresponding invariant set.



(a) (b)

Figure 2.5: (a) Minimum robust invariant set for agent 1 as a function of λ_2 with λ_1 fixed. (b) Maximum robust invariant set for agent 2 as a function of λ_2 with λ_1 fixed.

2.3.2 Application to a supply chain problem

In this section, we apply the proposed controller to a reduced version of the MIT beer game and compare the performance with other control schemes. The MIT beer game was developed by Jay Forrester in the late 1950's to show his managements students how oscillations arise in a supply chain, see for example [92]. A supply chain is the set of structures and processes used by an organization to provide a service or a good to a consumer. Typically three phenomena take place in supply chains flows: oscillation, amplification and phase lag. Due to material or informational delays in the flows of the supply chain, they are prone to oscillation; that is, production and inventories overshoot and undershoot the optimal levels. The magnitude of the fluctuations increase as they propagate from the customer to the factory, with each upstream stage tending to lag behind the downstream one in what is commonly known as the bullwhip effect.

The original MIT beer game is composed of four agents: retailer, wholesaler, distributor and factory. Customers demand beer from the first one, who orders beer from the wholesaler, who orders and receives beer from the distributor, who finally orders and receives orders from the factory. There are shipping and processing delays at each stage. In [92], the original model and all the difficulties of the corresponding stock management problem are explained in detail. This problem has been widely used in the literature. In particular, in [23] it has been used as application example for a DMPC scheme. The main difference between the proposed scheme and the DMPC proposed in [23] is that in [23] the agents only communicated once and the only information shared was the future input trajectories (a strategy similar to Iter 1).

In this example, a reduced version of the problem with two agents is considered: the

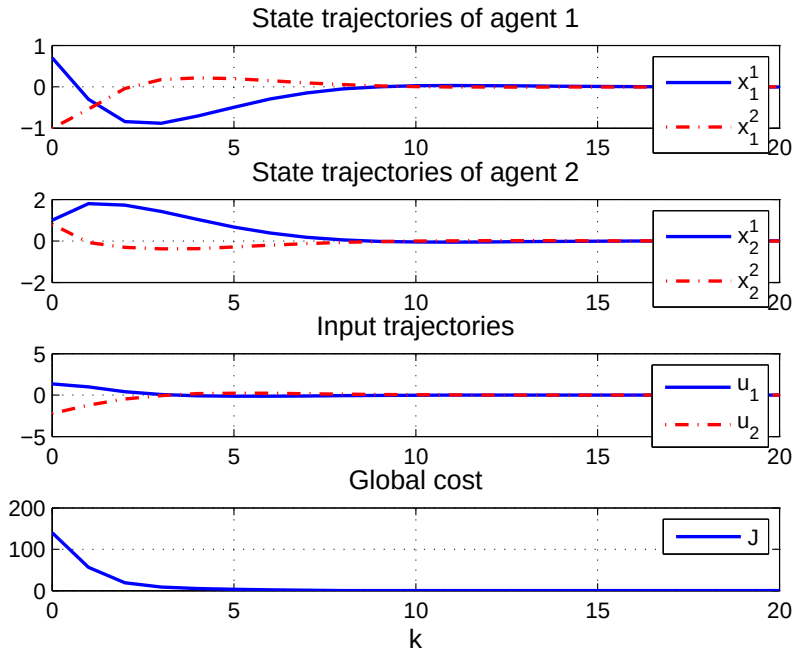


Figure 2.6: State, input and global cost $J(t)$ trajectories of the double integrators in closed-loop with the proposed controller.

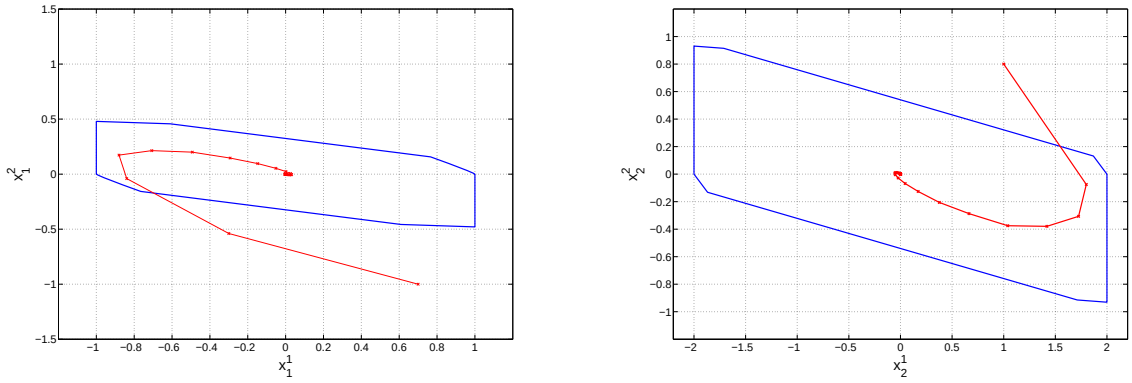
retailer and its supplier. There is no loss of generality since the structure of the game is regular: there is a cascade of firms, each maintaining and controlling its stock. The continuous time equations for the supplier are [92]:

$$\begin{aligned} \dot{s}^S(\tau) &= o_r^S(\tau-\tau_2) - o_r^R(\tau-\tau_1) - b^S(\tau)/t_b \\ \dot{o}_u^S(\tau) &= o_r^S(\tau) - o_r^S(\tau-\tau_2)) \\ \dot{b}^S(\tau) &= o_r^R(\tau) - o_r^R(\tau-\tau_1) - b^S(\tau)/t_b \end{aligned} \tag{2.13}$$

The equations for the retailer are:

$$\begin{aligned} \dot{s}^R(\tau) &= o_r^R(\tau-\tau_1-\tau_2) + b^S(\tau-\tau_2)/t_b - d_r(\tau-\tau_1) - b^R(\tau)/t_b \\ \dot{o}_u^R(\tau) &= o_r^R(\tau) - o_r^R(\tau-\tau_1-\tau_2) - b^S(\tau-\tau_2)/t_b \\ \dot{b}^R(\tau) &= d_r^R(\tau) - d_r^R(\tau-\tau_1) - b^R(\tau)/t_b \end{aligned} \tag{2.14}$$

The super-scripts R, S denote variables from the retailer and the supplier respectively. Variable $s^i(\tau)$ is the stock level; that is, the number of items available in that stage for shipment downstream. The unfulfilled order of stock $o_u^i(\tau)$ stands for the number of ordered items that the agent is waiting to receive from the upstream stage. The backlog of unfulfilled orders $b^i(\tau)$ accounts for the number of committed items that have to be shipped to the downstream stage. The parameter t_b stands for the average backlog clearance time and introduces a first



(a) (b)

Figure 2.7: (a) Agent 1 state evolution. (b) Agent 2 state evolution.

order dynamic in the process. The customer demand $d_r^R(\tau)$ represents how many items are demanded by the customers. From a control point of view, it can be seen as a measurable perturbation that has to be rejected in order to maintain the stock and the production at the desired levels. The information flows are assumed to have no time delays and the material flows have a delay modeled by τ_2. A delay for processing the received orders is introduced by means of the parameter τ_1. The manipulated variable at each stage is the order rate o_r^i; that is, the number of items demanded upstream. The supplier demands directly to the factory, which is modeled here as a pure delay. This model is different from other supply chain models, in which each agent has to decide not only what to order downstream, but what to send upstream. In this model of the MIT beer game, items sent to the upstream agent are not a decision variable. They are fixed by the orders received. In particular, items sent and the orders received are related through a first order system with a delay; that is, the shipment rate $l_r^i(\tau)$ is defined by the following equations

$$l_r^S(\tau) = o_r^R(\tau - \tau_1) + b^S(\tau)/t_b$$

$$l_r^R(\tau) = d_r^R(\tau - \tau_1) + b^R(\tau)/t_b$$

These relations have already been taken into account in the model.

The model of the system defined by the parameters τ_1, τ_2 and t_b. In the simulations performed we use $\tau_1 = 2d$, $\tau_2 = 1d$ and $t_b = 4d$. In order to obtain a discrete time model of the system, the continuous time model of equations (2.14)-(2.13) is discretized with a sampling time $\Delta = 1d$. Auxiliary states are introduced to take into account the delays. The resulting discrete time linear model is the one used in all the simulations carried out in this section.

In addition, an integrator is added to the controller; that is, the MPC controller decides

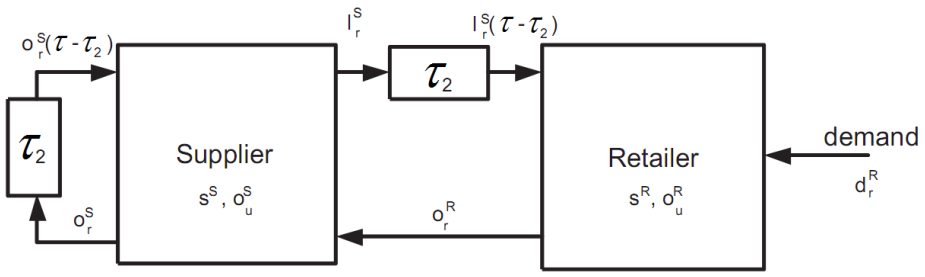


Figure 2.8: Reduced beer game.

the increment on the orders made downstream. This implies that the controller evaluates Δo_r^S and Δo_r^R defined as follows

$$\Delta o_r^S(t) = o_r^S(t) - o_r^S(t-1)$$
$$\Delta o_r^R(t) = o_r^R(t) - o_r^R(t-1)$$

The state of the model of the first subsystem (the retailer) is given by:

$$x_1(t) = \begin{bmatrix} s^R(t) \\ o_u^R(t) \\ b^R(t) \\ o_r^R(t-1) \\ o_r^R(t-2) \\ o_r^R(t-3) \\ b^S(t) \\ b^S(t-1) \\ d_r(t-1) \\ d_r(t-2) \end{bmatrix}$$

The state of the model of the second subsystem (the supplier) is given by:

$$x_2(t) = \begin{bmatrix} s^S(t) \\ o_u^S(t) \\ b^S(t) \\ o_r^S(t-1) \\ o_r^R(t-1) \\ o_r^R(t-2) \end{bmatrix}$$

It can be seen that both models share some information. In particular, the retailer model needs to keep track of the unfulfilled orders of the supplier, while the supplier model needs to

keep track of the orders received of the retailer. The input of the first agent is $u_1 = \Delta o_r^R(t)$ and the input of the second agent is $u_2 = \Delta o_r^S(t)$.

The control objective is to regulate the stock levels and the orders placed by both agents to a desired value. The orders received by the retailer from the external demand forces him to send items upstream, and hence to lose stock. These orders can be seen as external disturbances that have to be rejected. To this end, the retailer sends and order for more items downstream. These orders can be seen as an external disturbance for the supplier, which in order to reject this disturbances, generates new items. The retailer's stock has to be regulated to a reference value of $r_s^R(t)$. Analogously the supplier's stock is regulated to a value $r_s^S(t)$. The reference signals for the orders are given by $r_o^R(t)$ for the retailer and by $r_o^S(t)$ for the supplier. Note that in general, the orders references signal should be chosen accordingly with the predicted demand.

To this end, we consider different MPC controllers based on the following cost functions

$$J_1 = \sum_{k=0}^{N-1} (r_{s,k}^R - s_k^R)^2 W_s^R + (r_{o,k}^R - o_k^R)^2 W_o^R + (\Delta o_k^R)^2 W_\Delta^R$$
$$J_2 = \sum_{k=0}^{N-1} (r_{s,k}^S - s_k^S)^2 W_s^S + (r_{o,k}^S - o_k^S)^2 W_o^S + (\Delta o_k^S)^2 W_\Delta^S$$

where N is the prediction horizon, the subindex k denotes the k-steps predicted value of a signal and $W_s^R, W_o^R, W_\Delta^R, W_s^S, W_o^S, W_\Delta^S$ are constant weight matrices that define the stage cost. It is important to remark that no terminal cost function is considered in this example.

Note that in order to obtain predictions for the states of the retailer, an estimation of the future demand is needed. We denote the estimated demand as $\hat{d}_r^R(t)$. This signal may be different from the actual demand $d_r^R(t)$ in a given simulation.

The following values were used for the controller parameters:

$$N = 6, W_s^R = 30, W_o^R = 30, W_\Delta^R = 1, W_s^S = 30, W_o^S = 30, W_\Delta^S = 1 \tag{2.15}$$

For these simulations we have considered that the stocks and orders must be non negative.

The objective of this section is to compare the performance of different MPC schemes. To this end, we have carried out a set of simulations in five different scenarios for each controller. The first controller considered is the centralized MPC scheme defined by the optimization problem (2.6). This controller decides both inputs with a single optimization problem based on the full model of the system and the global cost function $J = J_1 + J_2$. In general the centralized MPC provides the best performance and has the higher computational burden. The second control scheme considered is the proposed distributed MPC controller in which two different agents communicate to find a cooperative solution. In addition, we have

considered several controllers based on the iterative controller defined by the optimization problems (2.7),(2.8). To avoid the case in which the agents do not reach an agreement, a maximum number of iterations is fixed. Different controllers with a maximum number of iterations of 1, 2, 5 and 10 have been considered. We denote these controllers as Iter1, Iter2, Iter5 and Iter10 respectively. In case of convergence in this bargaining process, the agents reach a Nash solution from a game theory point of view. None of them consider the cost function of the other agent. For any given input trajectory proposed by its neighbor, the agent evaluated the best possible input for his performance index. By definition, in a situation of equilibrium, this situation constitutes a Nash equilibrium. It is important to remark that in the controller defined by a single iteration (Iter1), the agents do not reach an agreement. They just advice each other about their predicted inputs. Each agent uses this information to estimate the future behavior of the other one. This is not a cooperative scheme because agents do not have a chance to bargain. From the point of view of each agent the other's actions are simply measurable disturbances.

In order to compare the performance of the controllers, four different scenarios have been taken into account. Each scenario is defined by a different initial state, a different retailer demand, and a different demand forecast. All the simulations are done with the discrete time model presented before.

Scenario 1: Both agents begin with 250 items in stock. The demand of the system $d_r^R(t)$ is defined the following way: during the first 15 days its value is 70. After that, it is set to 130 during 10 days and finally it returns to its initial value for 70 days. The estimated demand $\hat{d}_r^R(t)$ is equal to the real demand.

Scenario 2: Same initial state and estimated demand of the first scenario. In this case, the real demand differs from the estimated demand. At each time step, the real demand is obtained adding a random variable with mean 0 and standard deviation of 15 to the estimated demand.

Scenario 3: Same initial state and real demand of the first scenario. In this case the estimated demand is supposed to be constant and equal to the latest demand received; that is, the instant demand is extended in time as a forecast.

Scenario 4: Same real and estimated demands of the first scenario. The initial state is below the reference. The retailer has an initial stock of 100 items while the supplier is out of stock.

The results obtained are shown in tables 2, 3, 4 and 5. In these tables, the total accumulated cost and the total CPU time of each simulation is shown. The total CPU time includes not only the time of solving the different optimization problems but also all the addi-

Table 2.2: Results for scenario 1.

	J	T_{sim}
Centralized	3.6179e+006	1.4187
DMPC	4.9827e+006	0.8806
Iter1	2.1866e+007	0.5362
Iter2	5.6999e+006	0.7200
Iter5	5.8449e+006	1.7651
Iter10	4.1679e+006	2.2721

Table 2.3: Results for scenario 2.

	J	T_{sim}
Centralized	4.3228e+006	1.44
DMPC	5.3558e+006	0.8890
Iter1	2.1921e+007	0.5008
Iter2	6.0941e+006	0.8771
Iter5	6.2743e+006	1.8354
Iter10	4.7223e+006	2.2581

Table 2.4: Results for scenario 3.

	J	T_{sim}
Centralized	5.9327e+006	1.427
DMPC	9.6698e+006	0.8265
Iter1	2.2047e+007	0.6887
Iter2	9.0370e+006	0.8287
Iter5	1.0595e+007	1.8209
Iter10	6.2798e+006	1.2073

Table 2.5: Results for scenario 4.

	J	T_{sim}
Centralized	7.2608e+006	1.55
DMPC	8.1302e+006	0.9521
Iter1	2.9982e+007	0.6116
Iter2	1.0397e+007	0.8542
Iter5	1.0444e+007	1.8621
Iter10	9.4679e+006	2.4107

tional computations such as evaluating the system model. In the simulations the distributed schemes have not been implemented in parallel, and hence the centralized and the distributed controllers have the same computational power. The total simulation time provides an estimate of the computational burden of each of the controllers, in particular, it shows that for this particular example the centralized problem has a low computational burden and that the computational burden of the iterative controllers increase as the maximum number of iterations increase. In addition, for scenario 1 figures are shown for all the different controllers considered.

Some conclusions can be obtained from the preceding experiments. In general, the proposed algorithm provides a performance of the same order of magnitude than the one provided by the centralized MPC which, as expected, has the best results. Regarding the simulation time, it can be seen that for this particular case, the CPU time needed to solve in parallel the sequence of low order optimization problems is very similar to the time needed to solve the large scale problem. With respect to the non-cooperative distributed MPC controllers, the proposed distributed scheme provides a better performance than Iter1, Iter2 and Iter5. The controller Iter10 provides a better performance but needs more communication cycles in order to achieve an agreement. Even in this case, the solution is still a Nash equilibrium, so there is no guarantee that it will provide a good overall performance. Note that the iterative controllers results show that increasing the number of iterations of the bargaining process

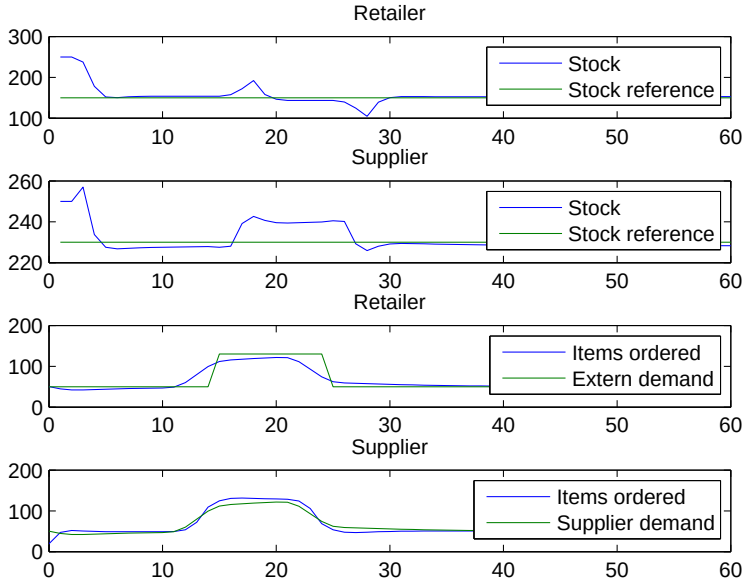


Figure 2.9: Centralized MPC closed-loop trajectories for scenario 1.

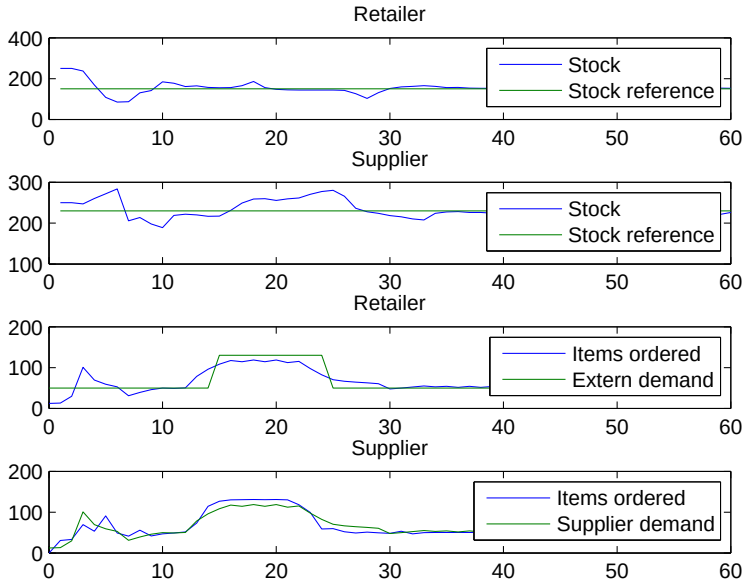


Figure 2.10: Proposed DMPC closed-loop trajectories for scenario 1.

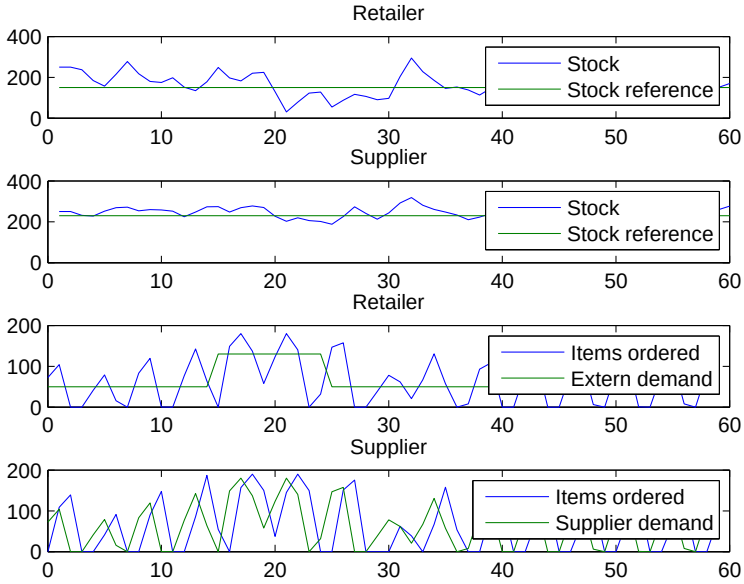


Figure 2.11: Iter1 closed-loop trajectories for scenario 1.

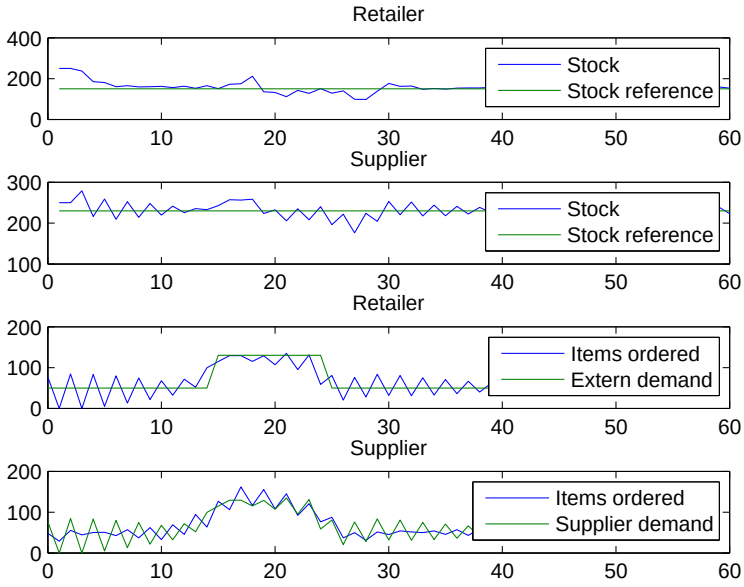


Figure 2.12: Iter2 closed-loop trajectories for scenario 1.

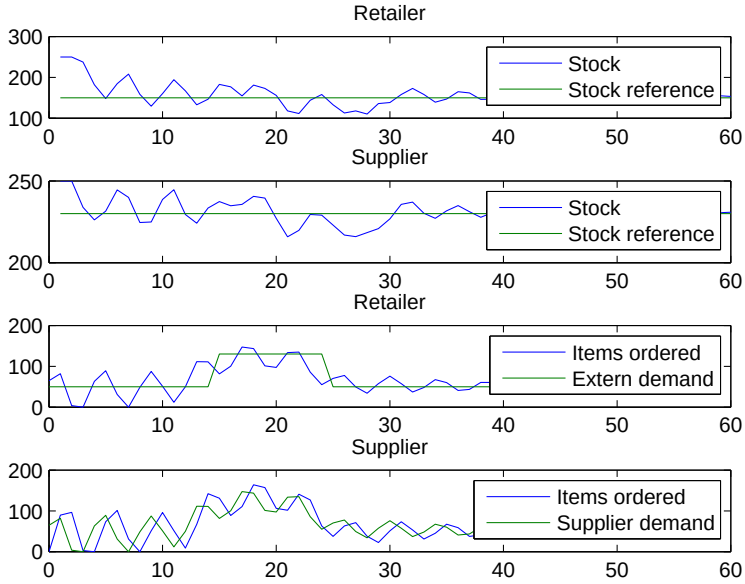


Figure 2.13: Iter5 closed-loop trajectories for scenario 1.

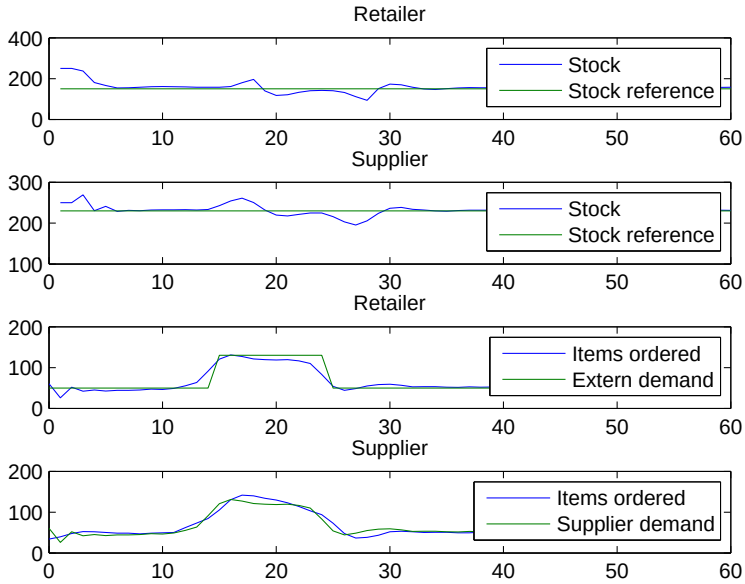


Figure 2.14: Iter10 closed-loop trajectories for scenario 1.

does not guarantee an improvement in the performance. It can be seen that Iter5 sometimes is worse than Iter2 from the performance point of view.

The simulations demonstrate that the proposed distributed scheme provides a good performance with only two communication cycles because it obtains a cooperative solution; that is, the decision is taken in order to optimize a global performance index. The iterative controllers do not take a cooperative decision and this implies, that in general, the solutions provided are worst. This can be clearly seen in the figures of scenario 1. In this scenario, the centralized MPC is able to react in advance to the demand peak, maintaining the stocks close to the references. The trajectories of the proposed distributed MPC scheme show a larger deviation of the stocks from the references, however, these trajectories do not present oscillations as the trajectories corresponding to Iter1, Iter2 and Iter5. Oscillations are a common result of non-cooperative bargaining processes. In this scenario, Iter10 however provides a better response that the proposed DMPC, at the cost of a high computational burden and a large number of communication steps.

2.4 Robustness of the proposed approach against data losses

In this section we carry out a set of simulations to study the robustness of the proposed approach when data losses occur in the control of a stirred tank reactor controller by two agents. The proposed algorithm assumes flawless communications between both agents. In a real distributed environment, errors in the communications and delays in the packets transmission should be expected. For simplicity, we will assume that an error in the communication link will affect the transmissions in both ways, so there is no possibility that only one of the agents is affected by the error. To test the effect of data losses in the closed-loop system performance,, we assume that the probability of flawless communications is given by the parameter $reliability \in [0, 1]$. This parameter characterizes the *quality* of the communication network. The higher it is, the better for the communication. In this section we show the results of simulations corresponding to different values of the parameter $reliability$.

In the original algorithm the agents chose among three options for the control signal (U_i^0, U_i^*, U_i^w) with the goal of minimizing J. When data losses occur, the agents do not receive U_i^w or the information needed to build the global cost table. In this case, each agent must decide whether to keep applying the last optimal input trajectory U_i^0, or behave selfishly and try to minimize its local cost function choosing U_i^*. In order to test the robustness of the proposed approach on the worst possible case, we assume that when communication errors occur each controller operates in a decentralized way, that is, applying U_i^*.

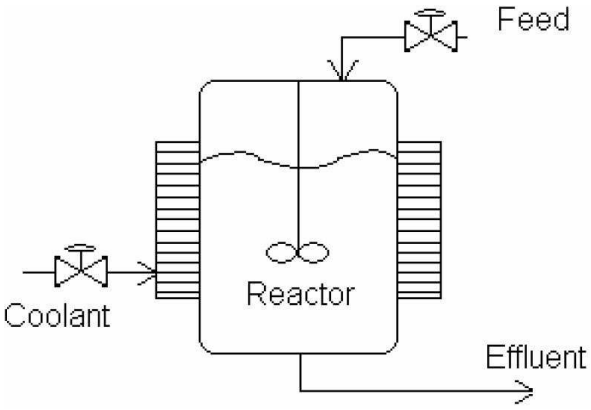


Figure 2.15: Continuously stirred tank reactor (CSTR)

Remark 1 *Note that as the parameter reliability tends to zero, the amount of information shared by the agents decreases, and the controller tends to operate in a decentralized manner.*

To demonstrate the robustness of the proposed controller against communication errors, we use the linearized model of a continuously stirred tank reactor (CSTR) presented in [12]. The linearized process around a given equilibrium point is described in continuous time by the following transfer matrix[2]

$$\begin{bmatrix} y_1(s) \\ y_2(s) \end{bmatrix} = \begin{bmatrix} \frac{1}{1+0.7s} & \frac{5}{1+0.3s} \\ \frac{1}{1+0.5s} & \frac{2}{1+0.5s} \end{bmatrix} \begin{bmatrix} u_1(s) \\ u_2(s) \end{bmatrix},$$

where the manipulated variables u_1 and u_2 are respectively the flow rate and the flow of coolant in the jacket. The controlled variables y_1 and y_2 are respectively the effluent concentration and the reactor temperature, see figure 2.18. The sampling time is defined as $T_s = 0.03s$.

The control objective is to track a given constant reference from a random initial state. We first design a centralized MPC scheme for comparison purposes. This controller decides both inputs simultaneously. For this reason, the MPC optimization problem that has been used for the simulations is based on minimizing the following cost function using the linearized discrete model of the process

$$\begin{aligned} J = \sum_{k=0}^{N-1} & (ref_1(k) - y_1(k))^T W_{y,1} (ref_1(k) - y_1(k)) \\ & + (ref_2(k) - y_2(k))^T W_{y,2} (ref_2(k) - y_2(k)) \\ & + \Delta u_1(k)^T W_{\Delta u,1} \Delta u_1(k) \\ & + \Delta u_2(k)^T W_{\Delta u,2} \Delta u_2(k) \end{aligned}$$

[2]The notation x(s) refers to the Laplace transform of the signal x.

where $ref_i(k)$ is the reference signal for the controlled variables y_1 and y_2. Note that the cost function depends on the predicted values of the inputs and outputs. In particular, $x(k)$ stands for the predicted value of the signal x k steps ahead in the future.

The following values were used for the controller parameters:

$$\begin{aligned} &N = 5 \\ &ref_1 = 0 \\ &ref_2 = 0 \\ &W_{y,1} = W_{y,2} = 1 \\ &W_{\Delta u,1} = W_{\Delta u,2} = 0.05 \end{aligned} \tag{2.16}$$

Note that we have not considered constraints on the input or the outputs in the simulations we have performed to test the robustness of the distributed scheme with respect to data losses.

In order to analyze the performance of our distributed scheme, we will make a comparison with a decentralized and a centralized controller. Note that the centralized controller provides the optimal solution from point of view of the performance while the decentralized one allows to determine what to expect in the absence of communication. The comparison will be made based on the following parameters:

- λ**: Convergence rate of the global cost function.** This parameter is computed as the smallest value λ such that the following inequality holds

$$J(kT_s) \leq J_0 \cdot \lambda^k, \lambda > 0 \tag{2.17}$$

  where $J(kT_s)$ is the value of the global cost function evaluated at time $t = kT_s$ for the applied input trajectories and $J(0)$ is its initial value, that is, at time $t = 0$. If $\lambda > 1$ the controlled system is unstable.

- t_r**: Rise time.** Number of sample times required in order to get a relative error below 5%, where the relative error is defined as

$$E_{ri} = \left| \frac{ref_i - y_i}{ref_i} \right| \cdot 100. \tag{2.18}$$

In first place, we will focus on the centralized controller. More than twenty simulations of the closed-loop system with the centralized controller were done, all of them beginning with different initial states. Half of the simulations were done for a number of $k_{max} = 100$ time

samples and the other half with $k_{max} = 300$. The average performance parameters obtained for the centralized controller were:

$$\lambda = 0.77$$
$$\text{Jv} = 13.$$

In order to apply decentralized and distributed MPC schemes, we considered that the CSTR is controlled by two different agents. Agent 1 controls the flow rate u_1 based on the measurements of the y_1, while agent 2 controls u_2 based on the measurements of y_2. Each agent has an incomplete model of the system; that is, they only know the first row of the system model (how their measured output is affected by each of the inputs). A decentralized MPC scheme is based on the idea that each agent tries to control its own subsystem without communicating with the other agent. Each agent tries to minimize a local cost function. For agent 1 the local cost function is

$$J_1 = \sum_{k=0}^{N-1} (ref_1(k) - y_1(k))^T W_{y,1}(ref_1(k) - y_1(k)) + \Delta u_1(k)^T W_{\Delta u,1} \Delta u_1(k)$$

and for agent 2 the local cost function is:

$$J_2 = \sum_{k=0}^{N-1} (ref_2(k) - y_2(k))^T W_{y,2}(ref_2(k) - y_2(k)) + \Delta u_2(k)^T W_{\Delta u,2} \Delta u_2(k).$$

At each time step, agent 1 receives y_1 and finds the optimal sequence of inputs such that J_1 is minimized assuming that $u_2 = 0$. Agent 2 follows the same protocol. For this particular system the decentralized controller is not able to stabilize the system. These simulations demonstrate that even a simple system may become unstable when the control agents are not coordinated by a proper scheme.

After carrying out the simulations of the centralized and decentralized controllers, we focus on the proposed DMPC scheme. As in the decentralized scheme, each agent tries to minimize its corresponding local cost function J_i. In this case, however, agents do communicate and try to minimize the sum of their optimization functions following the proposed DMPC scheme. In first place we consider the case in which there are no data losses or delays ($reliability = 1$). As it can be seen in figure 2.16 , the proposed controller scheme is able to stabilize the closed-loop system. We carried out over twenty simulations with different initial states and constant references. For this set of simulations the performance parameters were $t_r = 41.1458$ and $\lambda = 0.8858$. It can be seen that the performance of the distributed scheme is worst than the one of the centralized controller (although much better than the decentralized scheme which is not able to stabilize the system). As mentioned before, the centralized scheme is the best possible controller from the communication point of view.

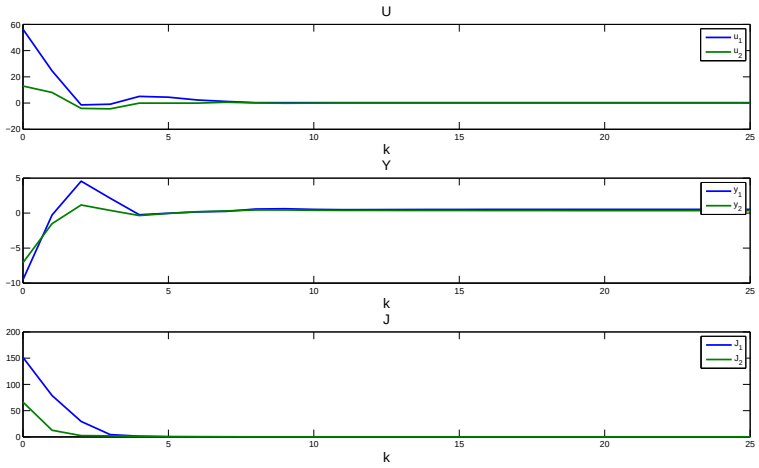


Figure 2.16: Trajectories of the system in closed-loop with the proposed DMPC and $reliability = 1$.

Table 2.6: λ and t_r for $reliability \in [0.1, 0.9]$.

	reliability								
	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1
$\lambda, k_{max} = 100$	0.89	0.89	0.90	0.93	0.99	1.17	1.39	1.60	1.93
$t_r, k_{max} = 100$	44.63	48.02	64.13	76.57	84.15	98.86	100	100	100
$\lambda, k_{max} = 300$	0.95	0.96	0.96	0.97	0.98	1.16	1.38	1.60	1.92
$t_r, k_{max} = 300$	45.37	52.81	71.31	90.93	129.62	291.03	300	300	300

In order to test the robustness of the proposed DMPC with respect to communications errors, a set of simulations with different *reliability* values were carried out. for each value *reliability* over 20 simulations were done. The results obtained are shown on table 2.4. Note that the value of k_{max} affects the value of the comparison parameters. An increment in the value of λ is found when k_{max} increases if the system stays stable ($\lambda < 1$). This is logical given the definition of the parameter λ. For example, if the system has reached the reference after k_1 sample times, the evolution during the rest of the time steps until the end of the simulation won't be significant. Thus, it can be concluded that the time samples after the system has reached the reference only degrade quantitatively the value of λ. Note too that t_r grows with k_{max}. Again, this is expected because when the controller is unable to regulate the system to the reference, the value $t_r = k_{max}$ is taken. Hence, it is obvious that the k_{max} has influence over t_r. Finally, note how the value of the performance parameters improves as the parameter *reliability* grows. In particular, it can be seen how the value tends to the one obtained with the flawless communication simulations. The simulation results also show that depending on

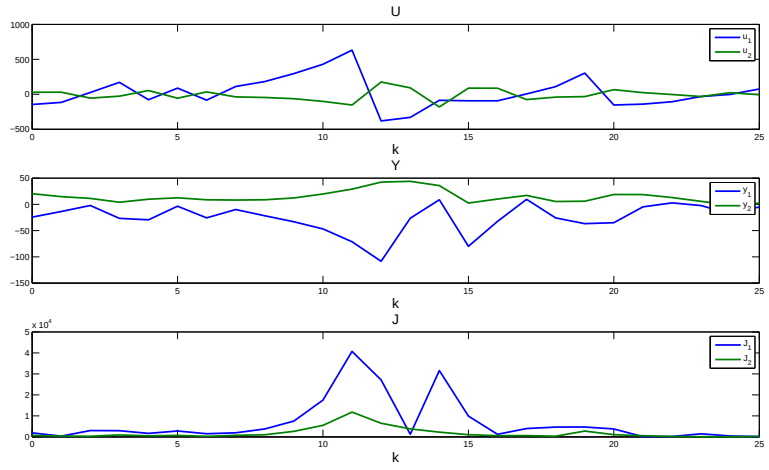


Figure 2.17: Trajectories of the system in closed-loop with the proposed DMPC and reliability = 0.5.

the value of *reliability*, the DMPC is able to stabilize the closed-loop system or not. For *reliability* ≥ 0.5, the performance parameter is $\lambda < 1$, which implies that the closed-loop system is stable. However, if more than 50% of the communications fails, then the DMPC is not able to stabilize the closed-loop system. These results show that when communication network becomes faulty, the proposed controller tends to operate in a decentralized manner, and hence, is not able to stabilize the system.

2.5 The four tank process

In this section we show experimental results of the proposed controller in a four tank process, which is one of the benchmarks of the european project HD-MPC. The physical plant is situated in facilities of the University of Seville and was presented in [6]. Different universities are working in the project as Delf Institute of Technology (Netherlands), Aachen University (Denmark) and Universidad Nacional de Colombia. In this chapter we present the results of the benchmark for the different DMPC policies developed by these universities. The results presented in this section have been submitted for publication in a joint work with the rest of the benchmark participants [5].

The use of benchmarks is useful for evaluating the capabilities of different approaches to control systems for real problems. Benchmarks allow to test, evaluate, and compare different control solutions at real or simulated plants. The research and the industry community

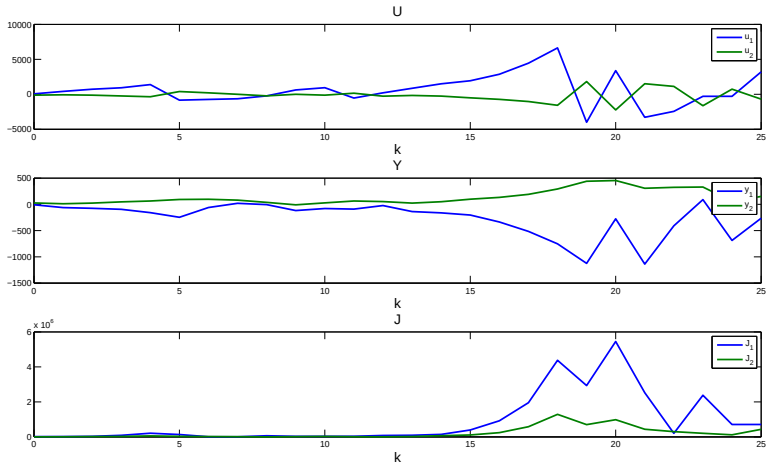


Figure 2.18: Evolution of the system with $reliability = 0.2$

benefit from these activities since the design of a good simulation test bed is often time and resource consuming. Furthermore, good simulation test beds are often subjected to heavy criticism as they either cover only a narrow part of the problem or they are purposely designed to get biased rather than objective performance results. The benchmark examples would effectively overcome these problems by: a) allowing an objective evaluation of control, b) reducing resources and time spent on developing validation models, c) giving researchers the possibility to evaluate their proposals on a variety of cases, and d) opening up a forum to compare the performance of various solutions and to discuss the quality of the results.

The four tank process has proven to be a very interesting system for control education and research [35]. The main characteristic of this process is that it is a simple multivariable system with highly coupled nonlinear dynamics that can exhibit transmission zeros dynamics. The four tank system has been used to illustrate advanced control strategies [19] such as internal model control and dynamic matrix control [25], multivariable robust control [95] and distributed MPC [62]. In addition, it has also been utilized as an educational tool in teaching advanced multivariable control techniques.

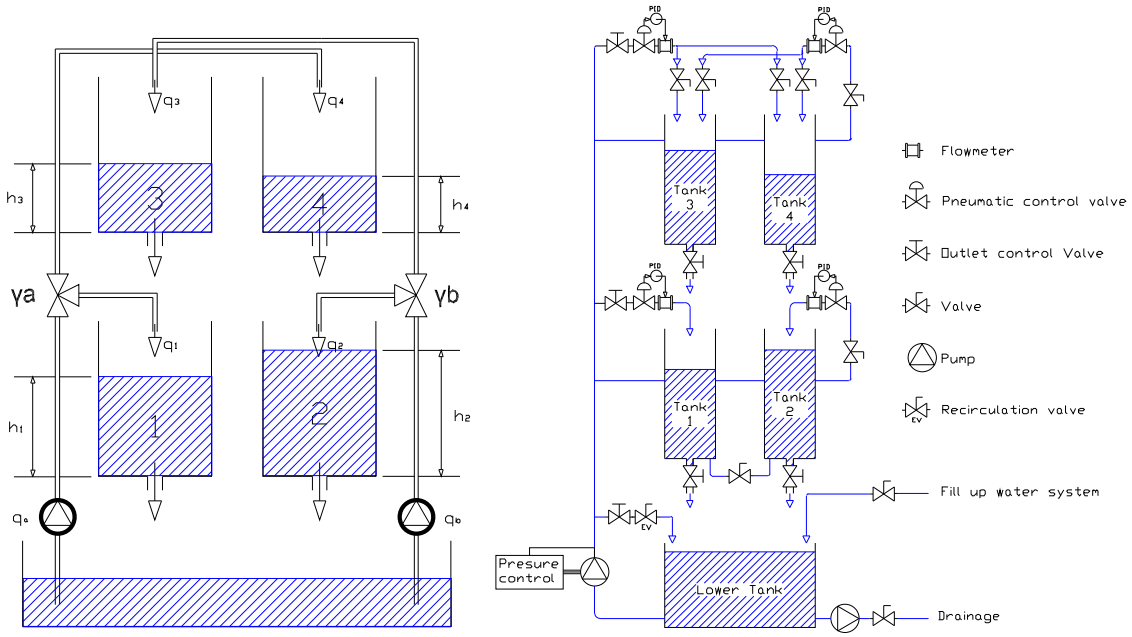
2.5.1 The four tank process

The four tank process that we have in the university of Seville is an educational plant designed to test control techniques using industrial instrumentation and control systems. The plant is a hydraulic process of four interconnected tanks inspired by the educational quadruple tank

process proposed by Johansson [35]. The main characteristic of this process is that it is a simple multivariable system with highly coupled nonlinear dynamics that can exhibit transmission zeros dynamics. The four tank plant retains the structure of Johansson's process (see Figure 2.20(a)), but has been modified to enable different configurations and interconnections of the tanks.

Figure 2.19: A photo of the four tank plant

A photograph of the plant can be seen in figure 2.19 and a schematic plot of the plant is shown in Figure 2.20(b). The inlet flow of each tank of the plant is measured by an electro-magnetic flow-meter (Siemens Sitrans FM Flowsensor 711/S and transmitters Intermag/transmag) and regulated by a pneumatic valve (Siemens VC 101 with a positioner Sipart PS2 PA). This allows the plant to emulate the three-ways valve of Johansson's quadruple tank process by providing suitable set-points to the flow controllers. The level of each tank is measured by means of a pressure sensor (Siemens Sitrans P 7MF4020 and 7MF4032). All the measurements and commands are 4-20 mA current signals and these are connected to a PLC (Siemens S7-200). The output flow of each tank can also be adjusted by means of a manual tuning valve which allows to adjust the speed of the dynamics of the plant. In order to achieve a safe operation of the plant and to prevent the overflow of tanks, each tank has a high level switching sensor used as alarm to switch off the pumps.



(a) Johansson's quadruple-tank process diagram.

(b) The real plant diagram.

Figure 2.20: The four tank process diagram.

Figure 2.20(a) shows the four tanks (T1, T2, T3 and T4), which are filled by several flows from a storage tank located at the bottom of the plant. The tanks at the top (T3 and T4) discharge in the tanks at the bottom (T1 and T2, respectively). The main valves regulate the flow of the main pipes of the plant. These are industrial control valves with an aperture controller which allows one to use it as a regulation valve or as a switching valve. The flow of each valve is continuously measured by a magnetic flow meter, allowing a flow control loop to manipulate the position of each valve.

The sampling of each sensor as well as the command of each manipulated variable is carried out by a Siemens PLC. This device stores the data and allows one to develop low level controllers (PIDs), sequential controllers and plant supervisors. All the data is continuously available by means of an OPC server installed in a remote PC connected to the PLC (via RS-232).

There are other several parameters of the plant that can be manually adjusted by the user (such as the section of the outlet hole a_i and the ratio of each three-ways valves). The discharge constant of each tank can be tuned by manipulating the regulation valve of its outlet. This regulation valve allows up to 40 different apertures of the valve. These apertures have been chosen to provide the maximal range of levels considering the maximum flow constraints of the plant. The three-way valves are emulated by a proper calculation of

the set-points of the flow control loops according to the considered ratio of the three-ways valve. Then, the manipulated variables of the plant can be considered the inlet flows of the three-ways valves q_a and q_b.

It is important to remark that the inlet flows and the levels of the tanks are physically limited (the values can be seen in table 2.7 in the following section). These limits must be taken into account in the controller design.

2.5.2 Four tank plant model

A continuous time state space model of the quadruple tank process system can be derived from first principles as follows [35]:

$$\begin{aligned}
\frac{dh_1}{dt} &= -\frac{a_1}{A_1}\sqrt{2gh_1} + \frac{a_3}{A_1}\sqrt{2gh_3} + \frac{\gamma_a}{A_1}q_a \\
\frac{dh_2}{dt} &= -\frac{a_2}{A_2}\sqrt{2gh_2} + \frac{a_4}{A_2}\sqrt{2gh_4} + \frac{\gamma_b}{A_2}q_b \\
\frac{dh_3}{dt} &= -\frac{a_3}{A_3}\sqrt{2gh_3} + \frac{(1-\gamma_b)}{A_3}q_b \\
\frac{dh_4}{dt} &= -\frac{a_4}{A_4}\sqrt{2gh_4} + \frac{(1-\gamma_a)}{A_4}q_a
\end{aligned} \tag{2.19}$$

where h_i, A_i and a_i with $i \in \{1, 2, 3, 4\}$ are the level cross section and the discharge constant of tank i, respectively; q_j and γ_j with $j \in \{a, b\}$ are the flow and the ratio of the three-ways valve of pump j, respectively; and g is the gravity. Throughout this section, the levels are measured in meters and the flows in cubic meters per hour.

Along the operation of the plant, it has been demonstrated that this model describes the plant dynamics very well, once the parameters (mainly the discharge constants of the tanks) have been identified, whenever the levels of the tanks are over 0.2 m. When the levels of the tanks are below 0.2 m, eddy effects in the discharge of the tank make the model become inaccurate. Therefore, when the levels are over 0.2 m, this model can be used to design the controllers guaranteeing that the derived controller will work similarly when controlling the real plant. For the control test presented in this section, the parameters of the plant are shown in table 2.7.

One important property of this plant is that the dynamics present multivariable transmission zeros which can be located in the right-hand side of the complex plane. In this benchmark, the values of γ_a and γ_b have been chosen in order to obtain a system with non-minimum phase multivariable zeros.

	Value	Unit	Description
H_{1max}	1.36	m	Maximum level of the tank 1
H_{2max}	1.36	m	Maximum level of the tank 2
H_{3max}	1.30	m	Maximum level of the tank 3
H_{4max}	1.30	m	Maximum level of the tank 4
H_{min}	0.2	m	Minimum level in all cases
Q_{amax}	3.26	m^3/h	Maximal flow of pump A
Q_{bmax}	4	m^3/h	Maximal flow of pump B
Q_{min}	0	m^3/h	Minimal flow
q_a^0	1.63	m^3/h	Equilibrium flow
q_b^0	2.0000	m^3/h	Equilibrium flow
a_1	1.310e-4	m^2	Discharge constant of tank 1
a_2	1.507e-4	m^2	Discharge constant of tank 2
a_3	9.267e-5	m^2	Discharge constant of tank 3
a_4	8.816e-5	m^2	Discharge constant of tank 4
A	0.06	m^2	Cross-section of all tanks
γ_a	0.3		Parameter of the 3-ways valve
γ_b	0.4		Parameter of the 3-ways valve
h_1^0	0.6487	m	Equilibrium level of tank 1
h_2^0	0.6639	m	Equilibrium level of tank 2
h_3^0	0.6498	m	Equilibrium level of tank 3
h_4^0	0.6592	m	Equilibrium level of tank 4

Table 2.7: Parameters of the plant

Linearizing the model at an operating point given by the equilibrium levels and flows shown in Table 2.7 and defining the deviation variables $x_i = h_i - h_i^0$, $u_j = q_j - q_j^0$ with $i \in \{1,2,3,4\}$ and $j \in \{a,b\}$ we obtain the following continuos time linear model:

$$\frac{dx}{dt} = \begin{bmatrix} \frac{-1}{\tau_1} & 0 & \frac{A_3}{A_1\tau_3} & 0 \\ 0 & \frac{-1}{\tau_2} & 0 & \frac{A_4}{A_2\tau_4} \\ 0 & 0 & \frac{-1}{\tau_3} & 0 \\ 0 & 0 & 0 & \frac{-1}{\tau_4} \end{bmatrix} x + \begin{bmatrix} \frac{\gamma_a}{A_1} & 0 \\ 0 & \frac{\gamma_b}{A_2} \\ 0 & \frac{(1-\gamma_b)}{A_3} \\ \frac{(1-\gamma_a)}{A_4} & 0 \end{bmatrix} u.$$

$$y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} x$$

where $\tau_i = \frac{A_i}{a_i}\sqrt{\frac{2h_i^0}{g}} \geq 0$, with $i \in \{1,2,3,4\}$ are the time constants of each tank. For the parameters of this benchmark it can be seen that the linear system is defined by four real stable poles and two non-minimum phase multivariable zeros.

2.5.3 The benchmark problem

The objective of the benchmark was to test and compare centralized, decentralized, and distributed predictive controllers under similar operation conditions. To this end the following experiment is defined in which the controllers must regulate the levels of tanks 1 and 2 to follow a set of reference changes by manipulating the inlet flows q_a and q_b based on the measured levels of the four tanks:

- The first set-points are set to $s_1 = 0.65$ m and $s_2 = 0.65$ m. This first reference is aimed to steer the plant to the operation point. Once the plant is in the operation point the test begins maintaining the operation point during 300 seconds.
- In the first step, the reference is changed to $s_1 = 0.3$ m and $s_2 = 0.3$ m during 3000 seconds.
- Then, the reference is changed to $s_1 = 0.5$ m and $s_2 = 0.75$ m during 3000 seconds.
- Finally, the set-points are changed to $s_1 = 0.9$ m and $s_2 = 0.75$ m during 3000 seconds. To perform this change tanks 3 and 4 have to be emptied and filled respectively.

The set-point signals are shown in Figure 2.21. The total control test takes 9300 seconds. The objective of the benchmark is to design the distributed MPC controllers to optimize the following performance index:

$$J = \sum_{i=0}^{N_{sim}} (h_1(i) - s_1(i))^2 + (h_2(i) - s_2(i))^2 + 0.01(q_a(i) - q_a^s(i))^2 + 0.01(q_b(i) - q_b^s(i))^2$$

where q_a^s and q_b^s are the steady manipulable variables of the plant for the set-points s_1 and s_2 calculated from steady conditions of the proposed model of the plant. Although the controllers tested have been designed using different sampling times, the performance index has been calculated for a sampling time of 5 seconds, that is, $N_{sim} = 1860$ samples.

The evaluation and comparison between the different controllers was performed according to a collection of indexes. These are aimed to compare different properties for the controllers as well as their behavior in the control test. These indexes are the following:

- Evaluation of the controller (`Qualitative Indexes`)
  1. Modelling requirements: the class of model considered by each of the controllers, for instance linear/nonlinear, plant model or subsystem model, etc.

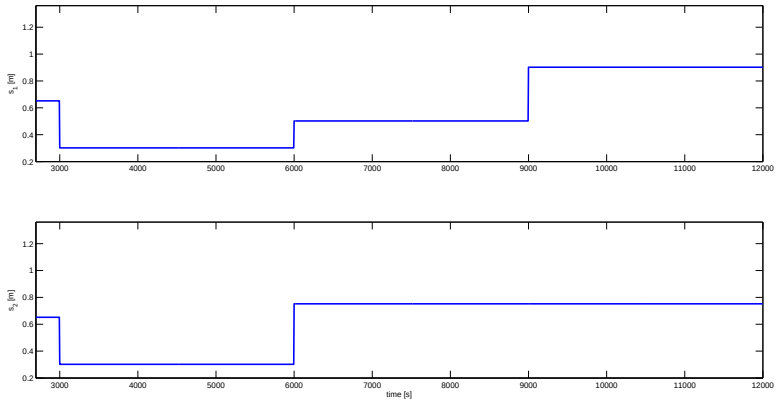


Figure 2.21: Set-point signals for the benchmark

2. Controller objectives: the properties addressed by the tested controllers, for instance optimality, constraint satisfaction, stabilizing design, recursive feasibility, etc.
3. Auxiliary software needed: optimization routines, simulation routines, etc.

- Evaluation of the test (`Quantitative`)

1. Performance index J: gives a measure of the performance of the whole trajectory of the controlled plant.
2. Performance index during the transient J_t: gives a measure of the performance index measured during the transients of the trajectory. This allows to remove the effect of the steady offset.
3. Settling time: gives a measure of the velocity of the controlled plant. This is calculated by summing all the settling times (for 95%) of the steps in the reference.
4. The number of floating point numbers in the data packet transmitted by the controllers: the total number of floating point numbers sent by one controller to the other during a sampling time.
5. Number of data packets transmitted during a sampling time: number of times that each controller sends data to the other controller.

The controller has been designed using a simulation model implemented in MATLAB/Simulink. Actually, each tested controller has been implemented using the same Simulink block. With a small modification this control model receives the measures and sends the calculated manipulable variables to the real plant by means of the OPC protocol. In the following section, the different control techniques are presented together with the results of the control test in the real plant.

2.5.4 Controllers under test

Application of the proposed controller

In first place we present the results of the distributed MPC scheme based on a cooperative game scheme presented in this chapter. In order to test the proposed DMPC scheme a discrete time linear model around the equilibrium point h_0, q_0 (which corresponds to the first reference) has been obtained linearizing the nonlinear model of the quadruple tank process with a sampling time of 5 seconds.

The state and input variables of the linearized model are defined as follows

$$x_1 = \begin{bmatrix} h_1 - h_{10} \\ h_3 - h_{30} \end{bmatrix}, u_1 = \begin{bmatrix} q_a - q_{a0} \end{bmatrix}, x_2 = \begin{bmatrix} h_2 - h_{20} \\ h_4 - h_{40} \end{bmatrix}, u_2 = \begin{bmatrix} q_b - q_{b0} \end{bmatrix}$$

The discrete linearized model of the first agent is

$$A_1 = \begin{bmatrix} 0.9705 & 0.0205 \\ 0 & 0.9792 \end{bmatrix}, B_{11} = \begin{bmatrix} 0.0068 \\ 0 \end{bmatrix}, B_{12} = \begin{bmatrix} 0.0001 \\ 0.0137 \end{bmatrix}$$

The model of the second agent is given by:

$$A_1 = \begin{bmatrix} 0.9661 & 0.0195 \\ 0 & 0.9802 \end{bmatrix}, B_{11} = \begin{bmatrix} 0.0002 \\ 0.016 \end{bmatrix}, B_{12} = \begin{bmatrix} 0.0091 \\ 0 \end{bmatrix}$$

The objective of the MPC controllers is to minimize a performance index that depends on the future evolution of both states and inputs based on the following local cost functions

$$J_1(x_1, U_1, U_2) = \sum_{j=1}^{N} (x_{1,j} - x_{1r})^T Q_1 (x_{1,j} - x_{1r}) + \sum_{j=0}^{N-1} R_1 (u_{1,j} - u_{1r})^2$$
$$J_2(x_2, U_2, U_1) = \sum_{j=1}^{N} (x_{2,j} - x_{2r})^T Q_2 (x_{2,j} - x_{2r}) + \sum_{j=0}^{N-1} R_2 (u_{2,j} - u_{2r})^2$$

where $N = 5$, $x_{i,j}$ and $u_{i,j}$ are the j-steps ahead predicted states and inputs of controller i respectively. The variables $x_{i,r}$ and $u_{i,r}$ are the target state and input obtained from the difference between the equilibrium point and the reference levels and flows. To determine these values, the nonlinear model has been used to obtain the levels of h_3, h_4 and the corresponding equilibrium flows q_a, q_b that guarantee that the references are an equilibrium point of the system. This implies that it has been done in a centralized manner. The controllers receive the appropriate references as inputs. In this point we have to remark the fact that when the reference is switched from one working point to another one it is necessary to reset

the value of U_s to a feasible solution. This solution is obtained solving a feasibility problem, in particular an LP, based on the full model of the system. Note that for this particular benchmark, no terminal region has been taken into account.

The weighting matrices were chosen to minimize the benchmark objective function, that is, $Q_1 = Q_2 = I$, $R_1 = R_2 = 0.01$. The local controller gains for each controller were $K_1 = [0.17\ 0.21]$ and $K_2 = [-0.16\ -0.14]$. These gains were designed with LMI techniques based on the full model of the system in order to stabilize both subsystems independently while assuring the stability of the centralized system. The role of these gains is important because the option in the game that allows to guarantee closed-loop stability is constructed shifting the last decided control action; that is, the first element is dropped after it is applied in the system and a term evaluated with these gains is added at the end of the horizon control vector, see [54] for more details.

The proposed distributed MPC controller only needs three communication steps in order to obtain a cooperative solution to the centralized optimization problem, has low communication and computational burdens and provides a feasible solution to the centralized problem. The simulation and experimental results show that the distributed scheme is able to control the system.

The designed controller has been sucessfully tested on the real plants and the trajectories are shown in Figure 2.22. The performance index of the test is $J = 29.5787$. The performance index is close to the performance index of the centralized MPC for regulation. Note however that the input trajectories are not smooth because the controllers switch between different modes during the simulation.

Other controllers

In the benchmark, the subsystems have been chosen according to the pairings derived from the relative gain array (RGA) analysis. Considering the values of the RGA the sensible pairing is to control the output h_1 with q_b (y_1 with u_2) and h_2 with q_a (y_2 with u_1). In first place, we mention the only decentralized controller that was tested. In particular a decentralized MPC for tracking was implemented, that is, a MPC for tracking [44] was designed for each subsystem. A communication based DMPC based on nonlinear dynamic optimization was tested as well. This controller is based on nonlinear dynamic optimization. The optimization is based on an iterative procedure of information broadcast in which the two local controllers exchange the value of the interaction variables. A more sophisticated cooperative version of the last algorithm was also tested. Concretely, the gradient-based distributed dynamic optimization (GBDDO) method [89] was put to test. Besides the communication of interac-

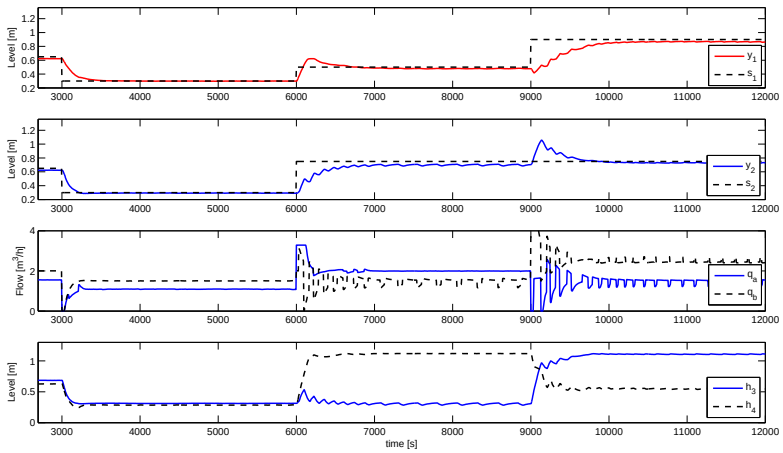


Figure 2.22: Evaluation of the control test in the real plant of the DMPC based on a cooperative game

tion variables, the GBDDO-based MPC requires a calculation and exchange of sensitivities. This information is used to modify the local cost function of each agent adding a linear term which partially allow to consider the other agents' objectives. The scheme proposed in [70, 72], which consists on a serial DMPC based on dual decomposition, was tested as well. This scheme is derived from a serial decomposition of an augmented Lagrangian formulation of the centralized overall MPC problem. Finally, it is remarkable that another controller based on game theory concepts was also implemented. Concretely, a feasible-cooperation distributed model predictive controller based on bargaining game theory concepts was tested. This approach models the DMPC as a game. According to [69], the solution of the cooperative game associated with the DMPC problem is given by a control vector which maximizes the product of the difference between the corresponding cost for each system and the cost of the subsystem when it does not cooperate with the rest[29, 76]. This last cost is computed through a minmax problem and is called disagreement point. The solution for this problem is computed in a distributed fashion following the same algorithm used in [98].

Evaluation of the controllers

Table 2.8 shows some qualitative properties of all the controllers that took part on the HDMPC benchmark. The entry *Model Requirements* shows whether the controllers need full or partial knowledge of the system and whether the model used is linear or nonlinear. The entry *Control Objectives* shows whether the controller is optimal from a centralized point of view (i.e., provides the same solution as the centralized MPC for regulation), guarantees

constraint satisfaction if a feasible solution is obtained and whether it can be designed to guarantee closed-loop stability in a regulation problem. The *Auxiliary software* entry shows which type of additional software is needed by each controller of the distributed scheme.

The centralized controller is based on a linear model of the full plant and is included as a reference for the performance of the distributed MPC schemes. On the other hand, the decentralized controller provides a reference on what can be achieved with no communication among the controllers at all. The other controllers assume that each controller has access only to its local state and model. All the controllers but the DMPC based on dynamic optimization are based on a linear model.

From the control objectives point of view, Table 2.8 shows whether the controller is optimal from a centralized point of view (i.e., provides an optimal solution to the corresponding centralized MPC), considers state and input constraints and whether it can be designed to guarantee closed-loop stability in a regulation problem. The decentralized controller considered cannot guarantee optimality, constraint satisfaction, nor stability. Note that in order to guarantee closed-loop stability, the the DMPC controller proposed in this thesis needs full model knowledge as we have mentioned in this chapter.

The distributed controllers that guarantee optimality (provided sufficient evaluation time) are the Serial DMPC and the DMPC based on dynamic optimization with GBDDO. Note that this controllers are also the ones with a larger communication and computational burden.

Another key issue in distributed schemes is the class of computational capabilities that each controller must have. In particular, for the schemes considered each controller must be able to solve either QP problems or general nonlinear optimization problems. In the experiments, the controllers used MATLAB's optimization toolbox, in particular `quadprog` and `fmincon`.

Evaluation of the experimental results

The experimental results demonstrated how a centralized solution provides the best performance while the performance of a fully decentralized controller is worse. Distributed schemes in which the controllers communicate in general improve this performance, although the experimental results also demonstrated that a distributed MPC scheme is not necessarily better than a decentralized scheme and it depends on the formulation of the controller and its design.

It is also clear how those controllers that incorporate offset-free techniques provide a better performance index. In order to obtain a measure of the performance without the effect of

Qualitative Indexes	Model Requirements	Control Objectives	Auxiliary Software
Centralized Regulation	Linear system Full model	Optimal Constraints Stability	QP
Decentralized	Linear system Local model	Suboptimal	QP
DMPC Coop. game	Linear system Local model (*Full model*)	Suboptimal Constraints (*Stability*)	QP
DMPC D.O. (GBDDO On)	Nonlinear system Local model	Optimal Constraints	NLP
DMPC D.O. (GBDDO Off)	Nonlinear system Local model	Suboptimal Constraints	NLP
DMPC Bargaining game	Linear system Local model	Suboptimal Constraints	NLP
Serial DMPC	Linear system Local model	Optimal Constraints	QP

Table 2.8: Table of qualitative benchmark indexes of each tested controller.

the steady offset, the transient performance index J_t has been calculated. This index is evaluated computing the cumulated cost during the transient. The entry t_s shows cumulated the settling time of the three reference changes. This shows that the offset-free controller (MPC for regulation) has a transient performance index similar to the total performance index while for the rest of the controllers, the transient index is better.

The effect of the communication between controllers on the calculation of the control inputs can be seen comparing the decentralized MPC with the DMPC based on a cooperative game. The size of the QPs to be solved at each sampling time is similar in both cases, while the computational time is larger in the DMPC case. The DMPC based on dynamic optimization exhibits the largest computational time due to the NLP to be solved at each iteration and the number of iterations.

All the controllers were implemented using a MATLAB function and were not designed to optimize the evaluation time. For this reason, the computation time has not been taken into account. In particular, both DMPCs based on dynamic optimization had a computation time lower than ten seconds, while the rest were of the order of one second. These computation times were lower than the sampling time chosen for each controller and moreover, they could be dramatically reduced using an appropriate implementation framework.

Motivated by these issues, the computational burden is best measured on the number and size of the optimization problems solved at each sampling time. The centralized schemes solve a single QP problem with $2N$ optimization variables while the decentralized controller solves 2 QP problems with N optimization variables. The difference in the computational burden between these schemes grows with the prediction horizon and the number of controllers. Distributed schemes try to find a tradeoff between computational/communicational burden and optimality. The DMPC based on a cooperative game and the DMPC based on a bargaining game solve a fixed number of low complexity optimization problems. DMPC based on dynamic optimization and Serial DMPC provide optimality at the cost of a higher computational burden.

In this particular benchmark, the best results are provided by the DMPC based on a cooperative game, however, there are several issues that must be taken into account. First of all, because the controller chooses among nine different modes of operation, the resulting input trajectories are not smooth. Figure 2.22 shows how the input seems to switch among at least two optimal trajectories. Depending on the application, this switching may not be acceptable. In addition, this control scheme is specially designed for only two controllers, because the number of possible modes grows in a combinatorial way with the number of controllers.

Quantitative Indexes	J	J_t	t_s	# floats	# trans
Centralized Regulation	25.46	23.78	2735	N.D	N.D.
Decentralized	39.54	21.2	1685	0	0
DMPC Coop. game	30.71	28.19	2410	20	3
DMPC D.O. (GBDDO On)	33.91	33.36	2555	150	5
DMPC D.O. (GBDDO Off)	35.65	34.63	1700	75	5
DMPC Bargaining game	46.32	39.52	3715	6	2
Serial DMPC	44.59	41.94	3130		

Table 2.9: Table of the quantitative benchmark indexes of each tested controller

2.6 Conclusions

In this chapter we have proposed a novel distributed MPC algorithm based on game theory for a class of systems controlled by two agents. The proposed controller only needs two communication steps in order to obtain a cooperative solution to the centralized optimization problem. Each agent solves an optimization problem that only depends on its local model and partial state information. After sharing information about the local cost, the agents choose the solution that yields the best global performance among a set of suboptimal possibilities. The options are suboptimal because each agent has an incomplete view of the system and they propose the best solutions from their point of view. The proposed algorithm has low communication and computational burdens and provides a feasible solution to the centralized problem. In addition, we provide sufficient conditions that guarantee practical stability of the closed-loop system as well as an optimization based procedure to design the controller so that these conditions are satisfied. Examples and real experiments have shown the properties the good performance of the controller, specially taking into account its low communicational and informational requirements. The robustness of the proposed scheme against failures in the communication channel has been proved as well, at least when the probability of failure is lower than a fifty percent. Finally, it is worthwhile to mention that the scheme has been tested and compared with other distributed algorithms in a benchmark of the european project HD-MPC.

Chapter 3

Distributed Model Predictive Control Based on Game Theory for Multiple Agents

In the previous chapter we presented a distributed scheme based on a cooperative game for the particular case in which the system is controlled by two agents. Unfortunately, the complexity of that scheme grows exponentially with the number of agents. In this chapter we propose a distributed model predictive control scheme based on agent negotiation suitable for problems in which the number of control agents is greater than two. Once more, we consider the control of several subsystems coupled through the inputs by a set of independent agents that are able to communicate and we assume that each agent has access only to the model and the state of one of the subsystems. This implies that in order to take a decision which is cooperative from a global point of view, i.e. for the whole system, the agents must negotiate. At each sampling time, following a given protocol, agents make proposals to improve an initial feasible solution on behalf of their local cost function, state and model. These proposals are accepted if the global cost improves the cost corresponding to the current solution. In addition, we study the stability properties of the proposed distributed controller and provide precise conditions based on a new concept of invariance for distributed and decentralized systems that guarantee that the closed-loop system is practically stable along with an optimization based controller and invariant design procedure. The theoretical results and the design procedure are illustrated using different simulation examples. In particular, we use an academical example to show the main theoretical contributions of this chapter. Next, we test the scalability of the distributed scheme with supply chains composed by an increasing number of nodes. The last example consists on the application of the proposed scheme to the control of irrigation canals.

The outline of the chapter is as follows. In section 3.1 the problem is formulated. Section 3.2 defines the proposed DMPC controller. Stability is studied in section 3.3 and a design method is given in section 3.4. An academical example to show the theoretical properties of the controller is presented in section 3.5. Section 3.6 deals with the application of the controller to a supply chain problem and section 3.7 studies its application to an irrigation canal problem. Finally, conclusions and future work are presented in section 3.8.

This chapter is based on the results and ideas submitted for publication in [57, 102, 103, 58].

3.1 Problem formulation

We consider the following class of distributed linear systems in which there are M_x subsystems coupled with their neighbors through M_u inputs

$$x_i(t+1) = A_i x_i(t) + \sum_{j \in n_i} B_{ij} u_j(t) \tag{3.1}$$

where $x_i \in \mathbb{R}^{q_i}$ with $i = 1, \ldots, M_x$ are the states of each subsystem, and $u_j \in \mathbb{R}^{r_j}$ with $j = 1, \ldots, M_u$ are the different inputs[1]. The set of indices n_i indicates the set of inputs u_j which affect the state x_i and the set of indices m_j indicates the set of states x_i affected by the input u_j. We define mathematically the concept of neighborhood of agent i as

$$N_i := \bigcup_{j \in n_i} m_j. \tag{3.2}$$

Therefore, any agent j included in N_i is a neighbor of agent i. Note that this does not imply that $i \in N_j$, that is, the neighborhood is not a symmetrical property in this context.

We consider the following linear constraints in the states and the inputs

$$\begin{array}{l} x_i \in \mathcal{X}_i,\ i = 1, \ldots, M_x \\ u_j \in \mathcal{U}_j,\ j = 1, \ldots, M_u \end{array} \tag{3.3}$$

where $\mathcal{X}_i$ and $\mathcal{U}_j$ are closed polyhedra that contain the origin in their interior defined by the following set of linear inequalities

$$\begin{array}{l} x_i \in \mathcal{X}_i \leftrightarrow H_{x_i} x_i \leq b_{x_i},\ i = 1, \ldots, M_x \\ u_j \in \mathcal{U}_j \leftrightarrow H_{u_j} u_j \leq b_{u_j},\ j = 1, \ldots, M_u \end{array} \tag{3.4}$$

Note that, as these polyhedra contain the origin in their interior, then $b_{x_i} > 0$ and $b_{u_j} > 0$.

[1] Throughout this chapter the time dependence is omitted when possible for notational convenience.

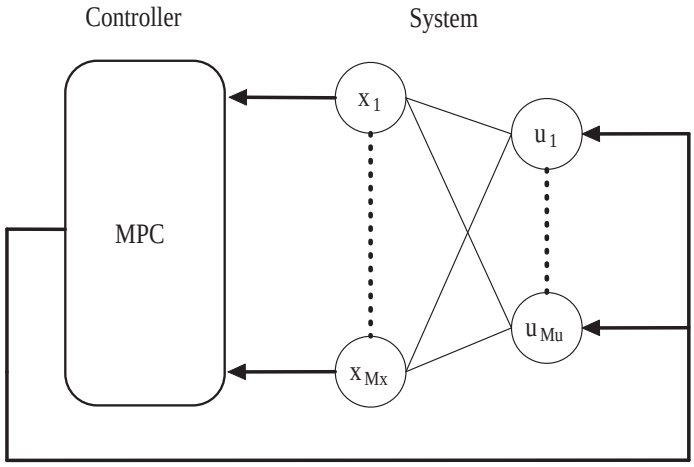


Figure 3.1: Centralized MPC.

There are many different physical systems that can be modeled under this formulation. For example, in [14] this model is used to represent the dynamics of a traffic network. In [70] the dynamics of an irrigation canal system are described with a similar formulation. In [54] the beer game, a typical supply chain problem, is described likewise.

This class of systems can be represented by a graph in which to each node either the state of one of the subsystems or one of the inputs available is assigned, and the arcs connect the inputs to the states they affect.

The control objective is to regulate the states of all the subsystems to the origin while satisfying the state and input constraints. To this end, centralized MPC follows a receding horizon approach and at each sampling time obtains the current states and solves a single finite horizon optimal control problem based on a performance index that depends on all the states and inputs. See figure 3.1 for a scheme of a centralized MPC controller. In distributed MPC schemes there are several agents that decide all the control inputs. It can be seen that although the states are not dynamically coupled, the agents need to negotiate in order to decide the value of the shared inputs. There are many possible distributed schemes depending on the available information and communication constraints. Figure 3.2 shows a scheme of a distributed controller in which each agent has access to partial state information and can communicate with the rest of the agents. This is the class of distributed control scheme considered in this work that is presented in the next section.

Remark 2 *One of the differences between the proposed approach and other cooperative MPC schemes is that the agents do not have a global model of the system. This may be important in some applications in which the centralized model is not available or the agents do not want to*

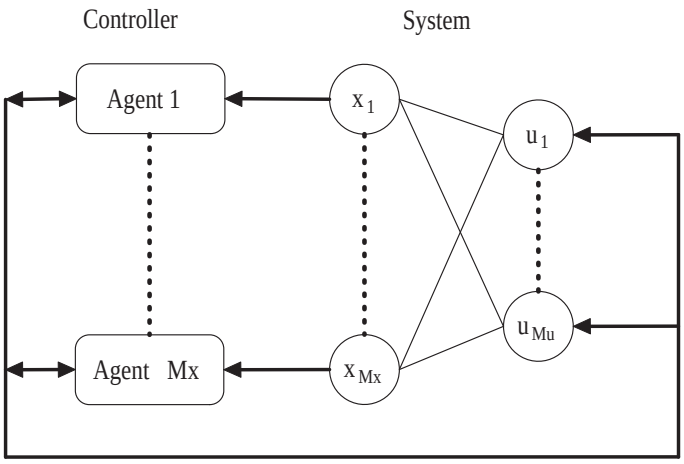


Figure 3.2: Distributed MPC.

share this information with the rest of the subsystems. In addition, there is a potential benefit from this assumption because if a distributed system adds a new subsystem, in the proposed scheme, only those agents affected by this new element would have to be updated, while in other schemes based on global information, the information would have to be broadcasted. One class of systems in which these issues are relevant are transport networks and supply chains, where new consumers/suppliers can appear dynamically.

Remark 3 *In the proposed scheme, several agents decide upon all or a subset of the control inputs. This implies that the inputs are not assigned to a particular agent as in most distributed MPC schemes found in the literature. Moreover, nothing is said about the magnitudes of M_x and M_u, thus this framework allows modeling situations in which there are agents with no associated inputs or even states. Hierarchical control or the existence of mediators in the network (agents that suggest an actuation for the rest of agents based on their own knowledge of the system) are examples of other interesting possibilities that can be also modeled with this framework.*

3.2 Proposed DMPC controller

In this chapter we propose a distributed scheme assuming that for each subsystem, there is an agent that has access to the model and the state of that subsystem. The agents do not have any knowledge of the dynamics of any of its neighbors, but can communicate freely among them in order to reach an agreement. The proposed strategy is based on negotiation

between the agents. At each sampling time, following a given protocol, agents make proposals to improve an initial feasible solution on behalf of their local cost function, state and model. These proposals are accepted if the global cost improves the cost corresponding to the current solution. To this end, the agent that makes the proposal must communicate with the neighbors affected. Note that a proposal may modify only a subset of inputs, and hence there are agents that may not be affected by these changes. Different negotiation/communication protocols may be implemented. The only requirement is that the protocol must guarantee that each proposal is evaluated independently. In this chapter, we propose to implement a controller in which at each sampling time, a fixed number of proposals made sequentially by random agents are considered.

The control objective of the proposed scheme is to minimize a global performance index defined as the sum of each of the local cost functions. The local cost function of agent i based on the predicted trajectories of its state and inputs defined as

$$J_i(x_i, \{U_j\}_{j\in n_i}) = \sum_{k=0}^{N-1} L_i(x_{i,k}, \{u_{j,k}\}_{j\in n_i}) + F_i(x_{i,N}) \tag{3.5}$$

where $U_j = \{u_{j,k}\}$ is the future trajectory of input j, N is the prediction horizon, $L_i(\cdot)$ with $i \in M_x$ is the stage cost function defined as

$$L_i(x_i, \{u_j\}_{j\in n_i}) = x_i^T Q_i x_i + \textstyle\sum_{j\in n_i} u_j^T R_{ij} u_j \tag{3.6}$$

with $Q_i > 0, R_{ij} > 0$ and $F_i(\cdot)$ is the terminal cost defined as

$$F_i(x_i) = x_i^T P_i x_i \tag{3.7}$$

with $P_i > 0$. We use the notation $x_{i,k}$ to denote the state i, k-steps in the future obtained from the initial state x_i applying the input trajectories defined by $\{U_j\}_{j\in n_i}$. Note that each of the local cost functions only depends on the trajectories of its state and the inputs that affect it.

At the end of the negotiation rounds, the agents decide a set of input trajectories denoted as U^d. The first input of these trajectories is applied, however, the rest of the trajectories are not discarded, instead are used to generate the initial proposal for the next sampling round which is given by the shifted future input trajectories U^s of all the inputs. The last input of each of these trajectories is given by

$$\sum_{p\in m_j} K_{jp} x_{p,N} \tag{3.8}$$

where $x_{p,N}$ is the predicted values of the state x_p after N time steps obtained applying $U^d(t-1)$ from the initial state $x_p(t)$. The set of shifted input trajectories will be applied in case the agents do not reach an agreement. This proposal is necessary in order to guarantee closed-loop stability.

We define next the proposed distributed MPC scheme:

- Step 1: Each agent p measures its current state $x_p(t)$. The agents communicate in order to obtain $U^s(t)$ from $U^d(t-1)$. In order to do this, each agent must receive $K_{ji}x_{i,N}$ from each agent i such that $K_{ji} \neq 0$ for some $j \in n_p$. The initial value for the decision control vector $U^d(t)$ is set to the value of the shifted input trajectories, that is, $U^d(t) = U^s(t)$.

- Step 2: Randomly, agents try to submit their proposals. To this end, each agent asks the neighbors affected if they are free to evaluate a proposal (each agent can only evaluate a proposal at any given time). If all the agents acknowledge the petition, the algorithm continues. If not, the agent waits a random time before trying again. We will use the superscript p to refer to the agent which is granted permission to make a proposal.

- Step 3: In order to generate its proposal, agent p minimizes J_p solving the following optimization problem:

$$\begin{aligned} \{U_j^p(t)\}_{j\in n_p} = \quad & \arg \min_{\{U_j\}_{j\in n_p}} J_p(x_p, \{U_j\}_{j\in n_p}) \\ & s.t. \\ & x_{p,k+1} = A_p x_{p,k} + \textstyle\sum_{j\in n_p} B_{pj} u_{j,k} \\ & x_{p,0} = x_i(t) \\ & x_{p,k} \in \mathcal{X}_p,\ k = 0, \dots N \\ & u_{j,k} \in \mathcal{U}_j,\ k = 0, \dots N-1,\ \forall j \in n_p \\ & x_{p,N} \in \Omega_p \\ & U_j = U_j^d(t),\ \forall j \notin n_{prop} \end{aligned} \tag{3.9}$$

  In this optimization problem, agent p optimizes over a set n_{prop} of inputs that affect its dynamics, that is, $n_{prop} \subseteq n_p$. Based on the optimal solution of this optimization problem, agent p presents a proposal defined by a set of input trajectories $\{U_j^p(t)\}_{j\in n_p}$ where $U_j^p(t)$ stands for the value of the trajectory of input j of the proposal of agent p. From the centralized point of view, the proposal at time step t of agent p is defined as

$$U^p(t) = \{U_j^p(t)\}_{j\in n_p} \biguplus U^d(t) \tag{3.10}$$

  where the operation $\biguplus$ stands for the update of the components relatives to $\{U_j^p(t)\}_{j\in n_p}$ in $U^d(t)$ and leaving the rest unmodified.

- Step 4: Each agent i who is affected by the proposal of agent p evaluates the predicted cost corresponding to proposed solution. To do so, the agent calculates the difference between the cost of the new proposal $U^p(t)$ and the cost of the current accepted proposal $U^d(t)$ as

$$\Delta J_i^p(t) = J_i(x_i(t), \{U_j^p(t)\}_{j\in n_i}) - J_i(x_i(t), \{U_j^d(t)\}_{j\in n_i}) \tag{3.11}$$

This difference $\Delta J_i^p(t)$ is sent back to the agent p. If the proposal does not satisfy the constraints of the corresponding local optimization problem, an infinite cost increment is assigned. This implies that unfeasible proposals will never be chosen.

- Step 5: Once agent p receives the local cost increments from each neighbor, it can evaluate the impact of its proposal $\Delta J^p(t)$, which is given by the following expression

$$\Delta J^p(t) = \sum_{i \in \bigcup_{j \in n_{prop}} m_j} \Delta J_i^p(t) \tag{3.12}$$

This global cost increment is used to make a cooperative decision on the future inputs trajectories. If $\Delta J^p(t)$ is negative, the agent will broadcast the update on the control actions involved in the proposal and the joint decision vector $U^d(t)$ will be updated to the value of $U^p(t)$, that is $U^d(t) = U^p(t)$. Else, is discarded.

- Step 6: The algorithm goes back to step 1 until the maximum number of proposals have been made or the sampling time ends. We denote the optimal cost corresponding to the decided inputs as

$$J(t) = \sum_{i=1}^{M_x} J_i(x_i(t), \{U_j^d(t)\}_{j \in n_i}) \tag{3.13}$$

- Step 7: The first input of each optimal sequence in $U^d(t)$ is applied and the procedure is repeated the next sampling time.

In figure 3.8 a flow diagram for a single agent of the proposed DMPC scheme is shown assuming that all the states are affected by all the inputs (hence, all the agents are neighbors). It can be seen that the agent must communicate several times with the rest of the agents. Note that in order to implement the proposed algorithm, it is necessary to obtain a set of future input trajectories that satisfy all the constraints for the initial state; that is, to initialize $U^s(0)$.

The situation that arises from the application of the proposed control strategy has been studied by game theory, the mathematical discipline that study all the phenomena that arise from the mutual interaction of agents that take their decisions alone or in cooperation [9, 66]. From a game theory point of view the situation can be described as a cooperative team game in which the possible strategies for each player are defined by its own proposals and the proposals of the rest of the agents. The utility of the proposals for each agent is defined by its local cost functions, however in order to find a solution, each agent chooses the option that is best from the global point of view.

Remark 4 *The time variable t, which is always used between parenthesis, references sampling times. The variable k, which is used always as a subscript, references the future time*

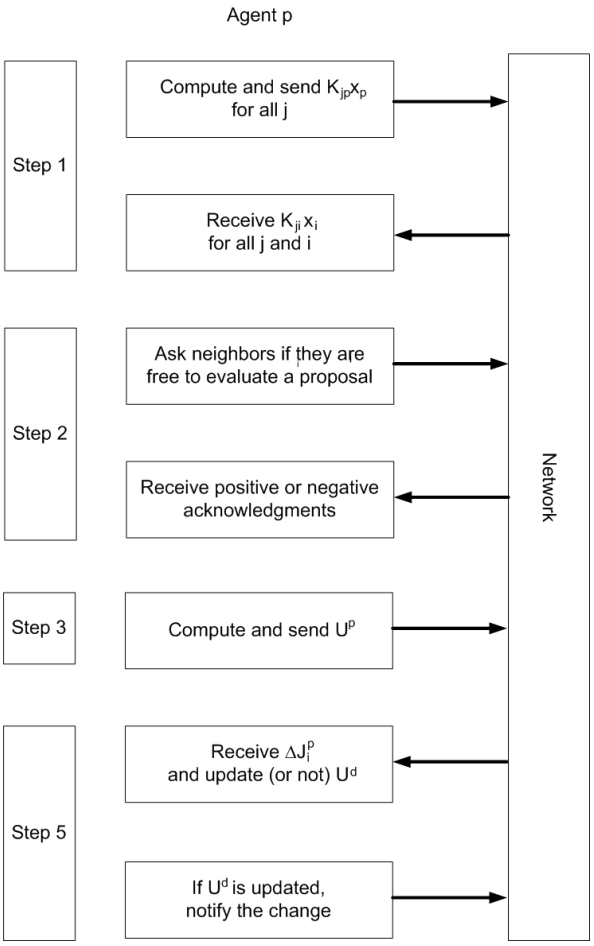


Figure 3.3: Flow diagram for a single agent which is granted permission to make a proposal of the proposed DMPC scheme.

steps along the prediction horizon of a given optimization problem and always takes values between 0 *and* N.

Remark 5 *Several proposals can be evaluated in parallel as long as they don't involve the same set of agents; that is, at any given time an agent can only evaluate a single proposal. The communication protocol to implement the algorithm in parallel is beyond the scope of this work.*

Remark 6 *Centralized MPC solves a single large-scale problem based on the model of the whole system such as the following optimization problem:*

$$\begin{aligned}\{U_j^c\}_{j=1,\dots,M_u} = \quad & \arg \min_{\{U_j\}_{j=1,\dots,M_u}} \textstyle\sum_{i=1}^{M_x} J_i(x_i, \{U_j\}_{j\in n_i}) \\ & s.t. \\ & x_{i,k+1} = A_i x_{i,k} + \textstyle\sum_{j\in n_i} B_{ij} u_{j,k} \\ & x_{i,0} = x_i \\ & x_{i,k} \in \mathcal{X}_i,\ k = 0,\dots,N \\ & u_{j,k} \in \mathcal{U}_j,\ k = 0,\dots N-1,\ \forall j \in n_i \\ & x_{i,N} \in \Omega_i \\ & \quad \forall i = 1,\dots,M_x\end{aligned} \tag{3.14}$$

3.3 Stability

Stability is a major issue in distributed systems. In general, it is a difficult problem because it is not enough to guarantee the stability of each of the subsystems. Actually, stable subsystems may lead to an unstable global system. In this section we provide sufficient conditions that guarantee asymptotic stability of the closed-loop system following a standard region/terminal cost approach [61].

Assumption 1 *There exist linear feedbacks* $u_j = \sum_{p\in m_j} K_{jp} x_p$ *and sets* $\Omega_i \subseteq R^{q_i}$ *such that if* $x_i \in \Omega_i$ *for all* $i = 1,\dots,M_x$ *then the following conditions hold for all* $i = 1,\dots,M_x$

$$\sum_{i=1}^{M_x} F_i(A_i x_i + \sum_{j\in n_i} B_{ij} \sum_{p\in m_j} K_{jp} x_p) - F_i(x_i) + L_i(x_i, \{\sum_{p\in m_j} K_{jp} x_p\}_{j\in n_i}) \le 0 \tag{3.15a}$$

$$A_i x_i + \sum_{j\in n_i} B_{ij} \sum_{p\in m_j} K_{jp} x_p \in \Omega_i \tag{3.15b}$$

$$\sum_{p\in m_j} K_{jp} x_p \in U_j \tag{3.15c}$$

$$\Omega_i \in X_i \tag{3.15d}$$

The requirements of Assumption 1 are twofold, first, the local feedbacks must satisfy constraint (3.15a) which implies that the system in closed-loop with these set of local controllers is stable. Second, sets Ω_i such that (3.15b) to (3.15d) are satisfied must exist. We denote these sets as jointly positive invariants for system (3.1) in closed-loop with the controllers defined by matrices K_{ij}. It is important to note that although the cartesian product of these sets is a positive invariant of system (3.1), in general it is not possible to obtain the jointly positive invariant sets from an invariant set of system (3.1) obtained following standard procedures because each Ω_i is defined only in a subspace of the whole state space; that is, in the space corresponding to the state x_i. This property is necessary in order to define for each agent a set of constraints that depend only on its state, and hence, only on its model. See the constraints of problem (3.9).

Theorem 4 *If Assumption 1 holds and at time step $t = 0$, $U^s(0)$ is given such that each of the M_x optimization problems (3.9)*[2] *are feasible for $x_{i,0} = x_i(0)$ and $U_j = U_j^s(0)$ with $i = 1, .., M_x$ and $j \in n_i$, then the proposed algorithm is feasible for all time steps $t \geq 0$ and system* (3.1) *in closed-loop with the proposed distributed MPC controller is asymptotically stable.*

Proof

The proof consists of two parts. We first prove that there is always a proposal which satisfies all the constraints (3.9) and then we prove that, under the stated assumptions,

$$J(t) = \sum_{i=1}^{M_x} J_i(x_i(t), \{U_j^d(t)\}_{j \in n_i}) \tag{3.16}$$

is decreasing sequence lower-bounded by zero.

Part 1. Taking into account that $\{U_j^d(t-1)\}_{j \in n_i}$ satisfies all the constraints of (3.9) and Assumption 1, it is easy to prove that $\{U_j^s(t)\}_{j \in n_i}$ provides a feasible solution for $x_i(t)$. It follows, that $U^d(t)$ provides a feasible solution for the optimization problem of agent i because it is chosen among a set of proposals which are required to be feasible in order to be accepted. Note that a proposal which is unfeasible for any of the agents cannot be chosen because the corresponding local cost is infinite. Taking into account that by assumption, $U^s(0)$ satisfies all the constraints for all the agents at time step $t = 0$ and using the above result recursively, the statement of this part is proved.

Part 2. Taking into account the definitions of $U_i^d(t-1)$ and $U_i^s(t)$ it follows that

$$J_i(x_i(t), \{U_j^s(t)\}_{j \in n_i}) - J_i(x_i(t-1), \{U_j^d(t-1)\}_{j \in n_i}) \tag{3.17}$$

[2] Although we used the index p in the definition of the optimization problems solved to obtain each proposal, in the proof of Theorem 4 we will use the index i.

is equal to

$$\begin{array}{l} F_i(A_i x_{i,N} + \sum_{j\in n_i} B_{ij} \sum_{p\in m_j} K_{jp} x_{p,N}) - F_i(x_{i,N}) \\ +L_i(x_{i,N}, \{\sum_{p\in m_j} K_{jp} x_{p,0}\}_{j\in n_i}) - L_i(x_{i,0}, \{\sum_{p\in m_j} K_{jp} x_{p,0}\}_{j\in n_i}) \end{array} \tag{3.18}$$

Taking into account (3.15a), this implies that

$$\sum_{i=1}^{M_x} J_i(x_i(t), \{U_j^s(t)\}_{j\in n_i}) - J(t-1) \leq -\sum_{i=1}^{M_x} L_i(x_{i,0}, \{\sum_{p\in m_j} K_{jp} x_{p,0}\}_{j\in n_i}) \tag{3.19}$$

As the proposed algorithm chooses $U^d(t)$ as an input trajectory that improves the cost, it is easy to see that

$$J(t) \leq J(t-1) - \sum_{i=1}^{M_x} L_i(x_{i,0}, \{\sum_{p\in m_j} K_{jp} x_{p,0}\}_{j\in n_i}) \tag{3.20}$$

Taking into account that recursive feasibility is guaranteed (see the first part of the proof) and the definitions of F_i and L_i and following the same lines of though as in [61] or [42], attractiveness and stability can also be proved. This implies that system (3.1) in closed-loop with the proposed distributed MPC controller is asymptotically stable.

■

The proof of Theorem 4 follows the standard terminal region/terminal constraint approach, see [61]. Stability is inherited from the set of local controllers defined by matrices K_{ij} which by (3.15a) are known to stabilize the system. In fact this result is based on the well known idea "Feasibility implies stability", see [90].

Remark 7 *The stability properties of the proposed scheme rely heavily on the fact that U_s satisfies all the constraints of the optimization problem. This implies, that in the start-up and when the controller looses feasibility due to disturbances, U_s has to be calculated either by a centralized supervisor or in a distributed manner by the agents.*

Remark 8 *When applied to a real system in the presence of disturbances and/or possible model errors, if the controller operates close to the state constraints in practice the shifted input trajectory may become unfeasible and in would have to be evaluated again (in a centralized manner or using an appropriate distributed approach). This issue must be taken into account in the implementation procedure of this control strategy.*

Remark 9 *Although in order to implement the proposed controller, the agents don't need information about the state or the dynamics of the rest of the subsystems, a centralized model of the full system is needed to design the controller so that closed-loop stability is guaranteed. This issue will be shown in the next section.*

3.4 Controller design procedure

The local controllers K_{ij} must satisfy two necessary conditions. First, the centralized system composed by the M_x subsystems (3.1) in closed-loop with the local controllers must be stable. Second, the jointly invariant sets must exist.

The local controllers that depend on each agent; that is, matrices K_{ji} such that $i \in m_j$, must be designed in a way such that (3.15a) holds. To take this condition into account, we will use the following centralized model of the system

$$x(t+1) = Ax(t) + Bu(t) \tag{3.21}$$

where

$$x = [x_1^T, \ldots, x_{M_x}]^T,\ u = [u_1^T, \ldots, u_{M_u}]^T \tag{3.22}$$

and matrices A and B are appropriate matrices that depend of the model (3.1) of each subsystem.

In addition, stability of each subsystem in closed-loop with its corresponding local feedback must be guaranteed. A sufficient condition to guarantee stability of each of the subsystems is to require that the cost function defined by the matrices P_i is a Lyapunov function for the subsystem in closed-loop with its corresponding local feedback. To take into account this condition, we will use the following uncertain model of each of the M_x subsystems

$$x_i(t+1) = A_i x_i(t) + B_i v_i(t) + E_i w_i(t) \tag{3.23}$$

where v_i is made of the part of the inputs that depend on x_i and w_i is the part of the inputs that depend on the rest of the states when the local controllers are applied; that is,

$$\begin{array}{l} B_i v_i(t) = \sum_{j \in n_i} B_{ij} K_{ji} x_i \\ E_i w_i(t) = \sum_{j \in n_i} B_{ij} \sum_{p \in m_j - \{i\}} K_{jp} x_p \end{array} \tag{3.24}$$

In this case, the objective is to design a controller $K_i = \{K_{ji}\}_{j \in n_i}$ that stabilizes the subsystem considering w_i an unknown disturbance. Matrices B_i and D_i are appropriate matrices that depend of the model (3.1) of each subsystem.

We provide next a set of linear matrix inequalities (LMI) that guarantees that (3.15a) holds and that K_i stabilizes the subsystem i. These LMI constraints are obtained following standard procedures, see for example [40, 1, 41].

Theorem 5 *Consider system (3.1). If there exist matrices W_i, Y_i with $i = 1, \ldots, M_x$ such*

that the following inequalities hold[3]

$$\begin{bmatrix} \Upsilon & \Phi & \Psi & \Xi \\ * & \Upsilon & 0 & 0 \\ * & * & I & 0 \\ * & * & * & I \end{bmatrix} \geq 0 \tag{3.25}$$

with $R_i = \sum_{j \in n_i} R_{ij}$, $R = diag(R_1, \ldots, R_{M_x})$, $K = [K_1, \ldots, K_{M_x}]$, $K_i^T = [K_{1i}, \ldots, K_{M_u i}]$ *and*

$$\Phi = \begin{bmatrix} W_1 A_1^T + Y_1^T B_1^T & Y_1^T B_2^T & \cdots & Y_1^T B_{M_x}^T \\ Y_2^T B_1^T & W_2 A_2^T + Y_2^T B_2^T & \cdots & Y_2^T B_{M_x}^T \\ \vdots & \vdots & \ddots & \vdots \\ Y_{M_x}^T B_1^T & Y_{M_x}^T B_2^T & \cdots & W_{M_x} A_{M_x}^T + Y_{M_x}^T B_{M_x}^T \end{bmatrix} \tag{3.26}$$

$$\Upsilon = \begin{bmatrix} W_1 & 0 & \cdots & 0 \\ * & W_2 & \cdots & 0 \\ * & * & \ddots & \vdots \\ * & * & * & W_{M_x} \end{bmatrix}, \Xi = \begin{bmatrix} Y_1^T R^{\frac{1}{2}} \\ Y_2^T R^{\frac{1}{2}} \\ \vdots \\ Y_{M_x}^T R^{\frac{1}{2}} \end{bmatrix}, \Psi = \begin{bmatrix} W_1 Q_1^{\frac{1}{2}} & 0 & \cdots & 0 \\ * & W_2 Q_2^{\frac{1}{2}} & \cdots & 0 \\ * & * & \ddots & \vdots \\ * & * & * & W_{M_x} Q_{M_x}^{\frac{1}{2}} \end{bmatrix} \tag{3.27}$$

and

$$\begin{bmatrix} W_i & W_i A_i^T - Y_i^T B_i^T & W_i Q_i^{\frac{1}{2}} & Y_i^T R_i^{\frac{1}{2}} \\ * & W_i & 0 & 0 \\ * & * & I & 0 \\ * & * & * & I \end{bmatrix} \geq 0 \tag{3.28}$$

for $i = 1, \ldots, M_x$ *then (3.15a) is satisfied for the matrices* $P_i = W_i^{-1}$, $K_i = \{K_{ji}\}_{j \in n_i} = Y_i W_i^{-1}$ *and systems (3.23) are stable in closed-loop with* $v_i = K_i x_i$.

Proof

We will prove the theorem in two parts. In the first part we will prove that if (3.25) holds, then (3.15a) is satisfied for the matrices $P_i = W_i^{-1}$, $K_i = \{K_{ji}\}_{j \in n_i} = Y_i W_i^{-1}$. In the second part, we will prove that if (3.28) holds then system (3.23) is stable in closed-loop with $v_i = K_i x_i$.

Part 1: In this part, we will prove that (3.25) is equivalent to (3.15a). Taking into account the definition of the centralized system (3.21), (3.15a) can be posed as follows

$$(A + BK)^T P (A + BK) - P + Q + K^T R K \leq 0 \tag{3.29}$$

[3]The symbol "$*$" stands for the symmetric part of a matrix.

with

$$\begin{aligned} R &= \text{diag}(R_1, \ldots, R_{M_x}) \\ Q &= \text{diag}(Q_1, \ldots, Q_{M_x}) \\ P &= \text{diag}(P_1, \ldots, P_{M_x}) \end{aligned} \tag{3.30}$$

with $R_i = \sum_{j \in n_i} R_{ij}$. Taking into account that the P and P^{-1} are positive defined matrices, if we multiply (3.29) by minus one and apply the Schur's complement we can recast (3.29) as the following constraint

$$\begin{bmatrix} P - Q - K^T R K & (A + BK)^T \\ (A + BK) & P^{-1} \end{bmatrix} \geq 0 \tag{3.31}$$

This LMI can be transformed into an equivalent one by pre and post multiplying it by a positive definite matrix

$$\begin{bmatrix} P^{-1} & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} P - Q - K^T R K & (A + BK)^T \\ (A + BK) & P^{-1} \end{bmatrix} \begin{bmatrix} P^{-1} & 0 \\ 0 & I \end{bmatrix} \geq 0 \tag{3.32}$$

The resulting equivalent matrix inequality is given by

$$\begin{bmatrix} P^{-1} - P^{-1} Q P^{-1} - P^{-1} K^T R K P^{-1} & P^{-1}(A + BK)^T \\ (A + BK)\, P^{-1} & P^{-1} \end{bmatrix} \geq 0 \tag{3.33}$$

In order to obtain a LMI inequality let $\Upsilon = P^{-1} = diag(W_1, W_2, \ldots, W_{M_x})$ with $W_i = P_i^{-1}$ for $i = 1, 2, .., M_x$ and $Y = K\Upsilon = [Y_1\ Y_2\ \ldots\ Y_{M_x}]$. It follows that

$$\begin{bmatrix} \Upsilon - \Upsilon Q \Upsilon - Y^T R Y & \Upsilon A^T + Y^T B^T \\ A\Upsilon + BY & \Upsilon \end{bmatrix} \geq 0 \tag{3.34}$$

Using the decomposition $Q = Q^{1/2} Q^{1/2}$ and applying Schur's complement we obtain

$$\begin{bmatrix} \Upsilon - Y^T R Y & \Upsilon A^T + Y^T B^T \\ A\Upsilon + BY & \Upsilon \end{bmatrix} - \begin{bmatrix} \Upsilon Q^{1/2} \\ 0 \end{bmatrix} I \begin{bmatrix} Q^{1/2}\Upsilon & 0 \end{bmatrix} \geq 0 \tag{3.35}$$

$$\begin{bmatrix} \Upsilon - Y^T R Y & \Upsilon A^T + Y^T B^T & \Upsilon Q^{1/2} \\ A\Upsilon + BY & \Upsilon & 0 \\ Q^{1/2}\Upsilon & 0 & I \end{bmatrix} \geq 0 \tag{3.36}$$

The same procedure is repeated for $R = R^{1/2} R^{1/2}$ obtaining

$$\begin{bmatrix} \Upsilon & \Upsilon A^T + Y^T B^T & \Upsilon Q^{1/2} \\ A\Upsilon + BY & \Upsilon & 0 \\ Q^{1/2}\Upsilon & 0 & I \end{bmatrix} - \begin{bmatrix} Y^T R^{1/2} \\ 0 \\ 0 \end{bmatrix} I \begin{bmatrix} R^{1/2} Y & 0 & 0 \end{bmatrix} \geq 0 \tag{3.37}$$

$$\begin{bmatrix} \Upsilon & \Upsilon A^T + Y^T B^T & \Upsilon Q^{1/2} & Y^T R^{1/2} \\ A\Upsilon + BY & \Upsilon & 0 & 0 \\ Q^{1/2}\Upsilon & 0 & I & 0 \\ R^{1/2} Y & 0 & 0 & I \end{bmatrix} \geq 0 \tag{3.38}$$

Defining $\Phi = \Upsilon A^T + Y^T B^T$, $\Psi = \Upsilon Q^{1/2}$ and $\Xi = Y^T R^{1/2}$, the following LMI constraint is obtained and hence the proof is completed

$$\begin{bmatrix} \Upsilon & \Phi & \Psi & \Xi \\ * & \Upsilon & 0 & 0 \\ * & * & I & 0 \\ * & * & * & I \end{bmatrix} \geq 0 \tag{3.39}$$

Part 2: In this part, we will prove that if (3.28) holds then system (3.23) is stable in closed-loop with $v_i = K_i x_i$. To this end, we will prove that (3.28) is equivalent to the following constraint

$$(A_i + B_i K_i)^T P_i (A_i + B_i K_i) - P_i + Q_i + K_i^T R_i K_i \leq 0 \tag{3.40}$$

which implies that $V_i(x) = x_i^T P_i x_i$ is a Lyapunov function of the closed-loop system and hence is stable. To prove this part of the theorem the constraint (3.40) is transformed in its equivalent LMI constraint (3.28) following the same procedure used in the first part.

■

Remark 10 *Additional constraints can be added to the design procedure so that there is no need to know the state x_i in order to calculate the input u_j. This is relevant because in order to evaluate the shifted input trajectory, all the subsystems whose state affects a given input must communicate, so in certain cases, it may be desirable to limit these communications.*

Once the local controllers and the terminal cost functions are fixed, in order to design a distributed MPC scheme that satisfies the assumptions of Theorem 4 one needs to find sets Ω_i such that (3.15b) to (3.15d) hold. In general this is a difficult problem because each of the sets depends on the others. The size of the terminal region for agent i is determined by the magnitude of the disturbances induced by its neighbor agents and viceversa. A similar class of invariant systems was studied in [81] within the polytopic games framework. We provide next an optimization based procedure to solve this problem. In order to present the algorithm we need the following definitions.

Definition 2 *Given the following discrete-time linear system subject to bounded additive uncertainties*

$$x^+ = \hat{A}x + \hat{B}u + \hat{E}w \tag{3.41}$$

with $w \in \hat{\mathcal{W}}$, subject to constraints in the state and the input $x \in \hat{\mathcal{X}}, u \in \hat{\mathcal{U}}$ and a linear feedback $u = \hat{K}x$; a set Ω is said to be a robust positive invariant set for the system if the

following constraints hold

$$\begin{aligned} & x \in \Omega \rightarrow (\hat{A} + \hat{B}\hat{K})x + \hat{E}w \in \Omega, \ \forall w \in \hat{\mathcal{W}} \\ & \hat{K}x \in \hat{\mathcal{U}} \\ & \Omega \subseteq \hat{\mathcal{X}} \end{aligned} \tag{3.42}$$

Given system matrices $\hat{A}, \hat{B}, \hat{E}, \hat{K}$ and the sets $\hat{\mathcal{X}}, \hat{\mathcal{U}}, \hat{\mathcal{W}}$, there exists several methods to find a set Ω that satisfies these constraints, see for example [39] for a procedure to find the maximal robust positive invariant and [80] for a procedure to find an approximation of the minimal robust positive invariant. We denote $\Omega(\hat{A}, \hat{B}, \hat{E}, \hat{\mathcal{X}}, \hat{K}, \hat{\mathcal{U}}, \hat{\mathcal{W}})$ the corresponding maximal robust positive invariant set.

In order to obtain sets Ω_i such that Assumption 1 is satisfied, we will use the uncertain model (3.23) of each agent; that is, each agent assumes that the contribution of its neighbors to the inputs that affect its dynamics are an unknown bounded disturbance. The size of the set in which these disturbances are bounded depend on the size of the sets Ω_i. This implies that finding these sets is in general a complex problem. In order to decouple the design of each set, each agent i limits its contribution to each input j by a factor $\lambda_{ji} \in (0, 1]$ with $\sum_{i \in m_j} \lambda_{ji} \leq 1$; that is,

$$K_{ji} x_i \in \lambda_{ji} \mathcal{U}_j, \ \forall i, j \tag{3.43}$$

Using the same notation introduced in (3.23), this implies that

$$v_i \in \mathcal{V}_i(\Lambda), \ w_i \in \mathcal{W}_i(\Lambda) \tag{3.44}$$

with

$$\begin{aligned} & V_i(\Lambda) = \lambda_{1i}\mathcal{U}_1 \times \lambda_{2i}\mathcal{U}_2 \times \ldots \times \lambda_{M_u i}\mathcal{U}_{M_u} \\ & W_i(\Lambda) = (\textstyle\sum_{p \in m_1 - \{i\}} \lambda_{1p})\mathcal{U}_1 \times (\sum_{p \in m_2 - \{i\}} \lambda_{2p})\mathcal{U}_2 \times \ldots \times (\sum_{p \in m_{M_u} - \{i\}} \lambda_{M_u p}\mathcal{U}_{M_u}) \end{aligned} \tag{3.45}$$

where $\Lambda = \{\lambda_{ij}\}_{\forall i,j}$ is a vector made of all the parameters λ_{ij}. Note that the maximum contribution of a given agent inside Ω_i, is the maximum contribution to the disturbance for the rest of the agents. In order to decouple the computation of the jointly invariant sets Ω_i, we use the following result based on finding a robust positive invariant set for each subsystem:

Lemma 1 *Given constants $\lambda_{ji} \in (0, 1)$ with $\sum_{i \in m_j} \lambda_{ji} \leq 1$, if the sets defined as*

$$\Omega_i = \Omega(A_i, B_i, E_i, \mathcal{X}_i, K_i, \mathcal{V}_i(\Lambda), \mathcal{W}_i(\Lambda)) \tag{3.46}$$

are not empty, they satisfy the constraints (3.15b) to (3.15d).

The lemma stems from the definition of the operator Ω. If all the sets exists, then they satisfy the stability constraints. Note that there exists an infinite number of possible values of λ_{ji} such that these sets exist. In order to chose one, we propose to solve the following optimization problem which maximizes the feasibility region of the distributed MPC controller:

$$\begin{aligned} \max_{\lambda_{ji}} \quad & f(\Omega_1 \times \Omega_2 .. \times \Omega_{M_x}) \\ & \Omega_i = \Omega(A_i, B_i, E_i, \mathcal{X}_i, K_i, \mathcal{V}_i(\Lambda), \mathcal{W}_i(\Lambda)) \\ & \lambda_{ji} \in (0,1),\ \forall j, i \\ & \textstyle\sum_{i \in m_j} \lambda_{ji} \leq 1,\ \forall i \end{aligned} \tag{3.47}$$

where function $f(\cdot)$ is a measure of the size of a polyhedron (for example, its Chebyshev radius).

Solving problem (3.47) may be difficult in general, however, under certain assumptions it can be posed as a convex problem. In [81] it was proved that the feasibility region of this problem is convex. In the next lemma we prove that the jointly invariant sets Ω_i are polyhedra defined by a set of inequalities whose right hand side can be expressed as an affine combination of the constants λ_{ij}. This implies, that if an appropriate function $f(\cdot)$ is chosen, problem (3.47) can be cast into a convex optimization problem.

Lemma 2 *If $A_i + \sum_j B_{ij} K_{ji}$ is stable, then the set*

$$\Omega_i = \Omega(A_i, B_i, E_i, \mathcal{X}_i, K_i, \mathcal{V}_i(\Lambda), \mathcal{W}_i(\Lambda)) \tag{3.48}$$

is a polyhedron that can be defined as a set of inequalities whose independent term can be expressed as an affine combination of the constants λ_{ij}, that is,

$$\Omega_i = \{x_i : M_i x_i \leq b_i + \sum_{j \in n_i} \sum_{p \in m_j} \lambda_{jp} b_{ij}\} \tag{3.49}$$

Proof:

The calculation of the robust invariant for a linear system is a well known problem and several procedures can be found in the literature, for instance in [37] or [39]. In order to prove the lemma, we will follow the procedure presented in [39]. The main idea is to find the set of states such that the trajectories of the closed-loop system starting from these states fulfill all the state and input constraints for all possible disturbances. This is done in an iterative manner. The set of states that fulfill the constraints after k steps is determined for increasing values of k. This process is repeated until convergence is obtained, that is, the same set of states is obtained for k and $k+1$. The resulting set is the maximum invariant set. Note that each value of k adds new constraints that the invariant set must fulfill, so the number of restrictions grows with each iteration.

First of all, we will define the state constraints that the closed-loop system has to satisfy taking into account the constraints in u_{ji}, the contribution of the state x_i to the different inputs u_j. By definition of Ω_i, u_{ji} has to verify

$$u_{ji} = K_{ji}x_i \in \lambda_{ji}\mathcal{U}_j,\ j \in n_i \tag{3.50}$$

Hence, the input constraint condition for the input j (3.4) can be transformed into the following set of inequalities:

$$H_{u_j}K_{ji}x_i \leq \lambda_{ji}b_{u_j},\ j \in n_i \tag{3.51}$$

Note that as $\lambda_{ji} \in (0,1)$, the set of inequalities is equal or more restrictive than the original input constraints. These inequalities have to be taken into account in the state constraints of the closed-loop system. The new set of state constraints can be written as

$$\hat{H}_{x_i}x_i \leq \hat{b}_{x_i} \tag{3.52}$$

For example, if $n_i = \{1, 2, \ldots, M_u\}$, that is, subsystem i is affected by all the inputs, then

$$\hat{H}_{x_i} = \begin{bmatrix} H_{x_i} \\ H_{u_1}K_{1i} \\ \vdots \\ H_{u_{M_u}}K_{M_ui} \end{bmatrix},\ \hat{b}_{x_i} = \begin{bmatrix} b_{x_i} \\ \lambda_{1i}b_{u_1} \\ \vdots \\ \lambda_{M_ui}b_{u_{M_u}} \end{bmatrix} \tag{3.53}$$

Note that the right hand side of the inequalities can be expressed as an affine combination of the constants λ_{ij} with $j \in n_i$.

Let $A_{CL_i} = (A_i + \sum_j B_{ij}K_{ji})$. If A_{CL_i} is stable, then for each value of Λ, the robust invariant set Ω_i can be determined in a finite number of steps backward $k(\Lambda)$. Let $k^* = \max_\Lambda k(\Lambda)$. We can compute the robust invariant set for all possible values of Λ as the set of states such that its k-steps ahead predictions satisfy all the constraints for all possible future disturbances; that is,

$$\hat{H}_{x_i}(A^k_{CL_i}x_i + \sum_{g=0}^{k-1} A^g_{CL_i} \sum_{j \in n_i} \sum_{p \in m_j - \{i\}} B_{ij}u_{jp}) \leq \hat{b}_{x_i},\ k = 1, \ldots, k^* \tag{3.54}$$

for all $u_{jp} \in \lambda_{jp}\mathcal{U}_j$ with $j \in n_i$ and $p \in m_j - \{i\}$. Taking into account that for all $u_{jp} \in \lambda_{jp}\mathcal{U}_j$ with $p \in m_j - \{i\}$ there exists $z_j \in \mathcal{U}_j$ such that

$$z_j \sum_{p \in m_j - \{i\}} \lambda_{jp} = \sum_{p \in m_j - \{i\}} u_{jp} \tag{3.55}$$

constraint (3.54) is equivalent to

$$\hat{H}_{x_i}(A^k_{CL_i}x_i + \sum_{g=0}^{k-1} A^g_{CL_i} \sum_{j \in n_i} B_{ij}z_j \sum_{p \in m_j - \{i\}} \lambda_{jp}) \leq \hat{b}_{x_i},\ k = 1, \ldots, k^* \tag{3.56}$$

for all $z_j \in \mathcal{U}_j$ with $j \in n_i$.

In order to eliminate the disturbance from the constraints and obtain a deterministic set, let us focus on each of the n_r rows of $\hat{H}_{x_i}$ (which define a different constraint for each time step k taken into account in the definition of the invariant set). To denote the r-th row of a matrix A we will use the $[A]_r$. Using this notation, constraint (3.56) is equivalent to

$$\begin{aligned}&[\hat{H}_{x_i} A^k_{CL_i}]_r x_i \leq [\hat{b}_{x_i}]_r - [\hat{H}_{x_i} \sum_{g=0}^{k-1} A^g_{CL_i} \sum_{j \in n_i} B_{ij} z_j \sum_{p \in m_j - \{i\}} \lambda_{jp})]_r, \\ &k = 1, \ldots, k^*,\ r = 1, \ldots, n_r\end{aligned} \tag{3.57}$$

Let us define

$$\sigma^{gr}_{ij} = max_{z_j \in \mathcal{U}_j}([(\hat{H}_{x_i} A^g_{CL_i} B_{ij})]_r z_j) \tag{3.58}$$

Note that σ^{gr}_{ij} is a scalar that can be calculated from the system model and constraints. This definition allows us to rewrite constraint (3.57) as:

$$[\hat{H}_{x_i} A^k_{CL_i}]_r x_i \leq [\hat{b}_{x_i}]_r - \sum_{g=0}^{k-1} \sum_{j \in n_i} \sigma^{gr}_{ij} (\sum_{p \in m_j - \{i\}} \lambda_{jp}),\ k = 1, \ldots, k^*,\ r = 1, \ldots, n_r \tag{3.59}$$

Taking into account that the second term of each of the constraints of (3.59) is an affine combination of the constants $\{\lambda_{ip}\}$ it is possible to find matrix M_i and vectors b_i and b_{ij} with $j \in n_i$ such that

$$\Omega_i = \{x_i : M_i x_i \leq b_i + \sum_{j \in n_i} \sum_{p \in m_j} \lambda_{jp} b_{ij}\} \tag{3.60}$$

■

Using this result, the problem of finding a matrix Λ that maximizes a measure of the distance can be cast into a convex optimization problem. For instance, let us suppose that our criterium to compare the invariant sets is the radium of a Chebyshev ball inside the invariant region. In this case we are interested in obtaining the maximum $x^T x$ as function of Λ that verifies all the constraints, which is a convex problem.

3.5 Example

Consider a system of the form (3.1) defined by the following matrices

$$\begin{aligned}
A_1 &= \begin{bmatrix} 1 & 0.8 \\ 0 & 0.7 \end{bmatrix}, B_{11} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, B_{12} = \begin{bmatrix} 0 \\ 0.15 \end{bmatrix}, B_{13} = \begin{bmatrix} 0 \\ 0.15 \end{bmatrix}, B_{14} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \\
A_2 &= \begin{bmatrix} 1 & 0.6 \\ 0 & 0.7 \end{bmatrix}, B_{21} = \begin{bmatrix} 0 \\ 0.15 \end{bmatrix}, B_{22} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, B_{23} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, B_{24} = \begin{bmatrix} 0 \\ 0.15 \end{bmatrix} \\
A_3 &= \begin{bmatrix} 1 & 0.9 \\ 0 & 0.8 \end{bmatrix}, B_{31} = \begin{bmatrix} 0 \\ 0.15 \end{bmatrix}, B_{32} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, B_{33} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, B_{34} = \begin{bmatrix} 0 \\ 0.15 \end{bmatrix} \\
A_4 &= \begin{bmatrix} 1 & 0.8 \\ 0 & 0.5 \end{bmatrix}, B_{41} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, B_{42} = \begin{bmatrix} 0 \\ 0.15 \end{bmatrix}, B_{43} = \begin{bmatrix} 0 \\ 0.15 \end{bmatrix}, B_{44} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}
\end{aligned} \tag{3.61}$$

subject to the following linear constraints in the state and the inputs

$$\begin{aligned}
&|x_1|_\infty \leq 1,\ |x_2|_\infty \leq 2,\ |x_3|_\infty \leq 1,\ |x_4|_\infty \leq 2 \\
&|u_1|_\infty \leq 1,\ |u_2|_\infty \leq 1,\ |u_3|_\infty \leq 1,\ |u_4|_\infty \leq 1
\end{aligned} \tag{3.62}$$

A graph that represents the couplings between the individual subsystems can be seen in figure 3.4. The box represent the subsystems while the arrows represent the coupling between neighbors. We assume that each agent can communicate with all the neighbors to evaluate the shifted input trajectory as well as the global cost of the proposals. The weighting matrixes that define the cost function of the MPC controller are the following:

$$Q_i = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, R_{ij} = 10 \tag{3.63}$$

with $i = \{1, 2, 3, 4\}$ and $j \in n_i$.

In order to implement the proposed DMPC control scheme we need to design the local feedbacks and the terminal cost functions according to LMI constraints presented in Theorem 5 to find matrices K_{ij} and P_i such that all the stability conditions are satisfied. In particular, matrices W, Y such that constraints (3.28) and (3.25) hold while maximizing the sum of the traces of the matrices W_i. Applying the variable change presented in Theorem 5,

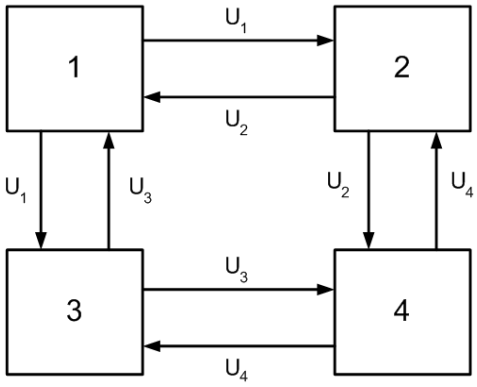


Figure 3.4: Graph of the system (3.61).

the following matrices K and P such that the stability assumptions hold are obtained

$$K^T = \begin{bmatrix} -0.27 & -0.01 & 0 & 0 \\ -0.59 & -0.02 & 0 & 0 \\ 0 & -0.28 & 0 & -0.01 \\ -0.01 & -0.5 & 0 & -0.02 \\ 0 & 0 & -0.24 & -0.01 \\ -0.01 & 0 & -0.68 & -0.02 \\ 0 & -0.01 & 0 & -0.30 \\ 0 & -0.02 & 0 & -0.48 \end{bmatrix}$$

$$P = \begin{bmatrix} 4.92 & 5.76 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5.76 & 11.30 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 5.65 & 5.42 & 0 & 0 & 0 & 0 \\ 0 & 0 & 5.42 & 8.82 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4.45 & 5.81 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5.81 & 13.74 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 5.61 & 5.80 \\ 0 & 0 & 0 & 0 & 0 & 0 & 5.80 & 8.95 \end{bmatrix} \tag{3.64}$$

The controller defined by matrix K stabilizes not only the centralized system but also the four subsystems individually considered. Note that in the optimization problem, additional constraints where imposed consisting in the absence of communication between some of the agents for the purpose of computing the local control law. This specification is reflected in the presence of zeros in the matrix. For example, agents 1 and 4 do not have to exchange any information in order to compute the shifted input trajectory. This class of additional constraints are particularly relevant when more involved communications protocols are taken into account.

The next step in the design procedure is to find the set of Λ that maximizes the size of the jointly invariant sets. In particular, we measure the size by the Chebyshev radius of the resulting centralized invariant set. The resulting optimization problem is convex problem (in particular, it can be posed as a LP problem) and has been solved using Matlab's `fmincon` function. The optimal matrix Λ is

$$\Lambda = \begin{bmatrix} 0.4568 & 0.0931 & 0.0115 & 0 \\ 0.0116 & 0.4576 & 0 & 0.0699 \\ 0.0805 & 0 & 0.4908 & 0.0235 \\ 0 & 0.1635 & 0.0354 & 0.4128 \end{bmatrix} \tag{3.65}$$

where the element of the i-th row and the j-th column corresponds to the constant λ_{ij}. Note that the constants λ_{ji} that correspond to matrices $K_{ji} = 0$ are set to zero.

The properties of the equivalent centralized system provide useful information to establish a comparison with the distributed approach. In particular, the size of the maximum invariant set for the centralized nominal case provides an upper bound of the size of the invariant set obtained from the jointly invariant sets. In this case, the radium of the largest Chebyshev ball is 0.74. The invariant set calculated for the distributed system has a radium of 0.66, a value very close to the centralized case. The reduction of the invariant region is 11%. In figure 3.5 the invariant set of each subsystem can be seen along with the corresponding projection of the centralized invariant set.

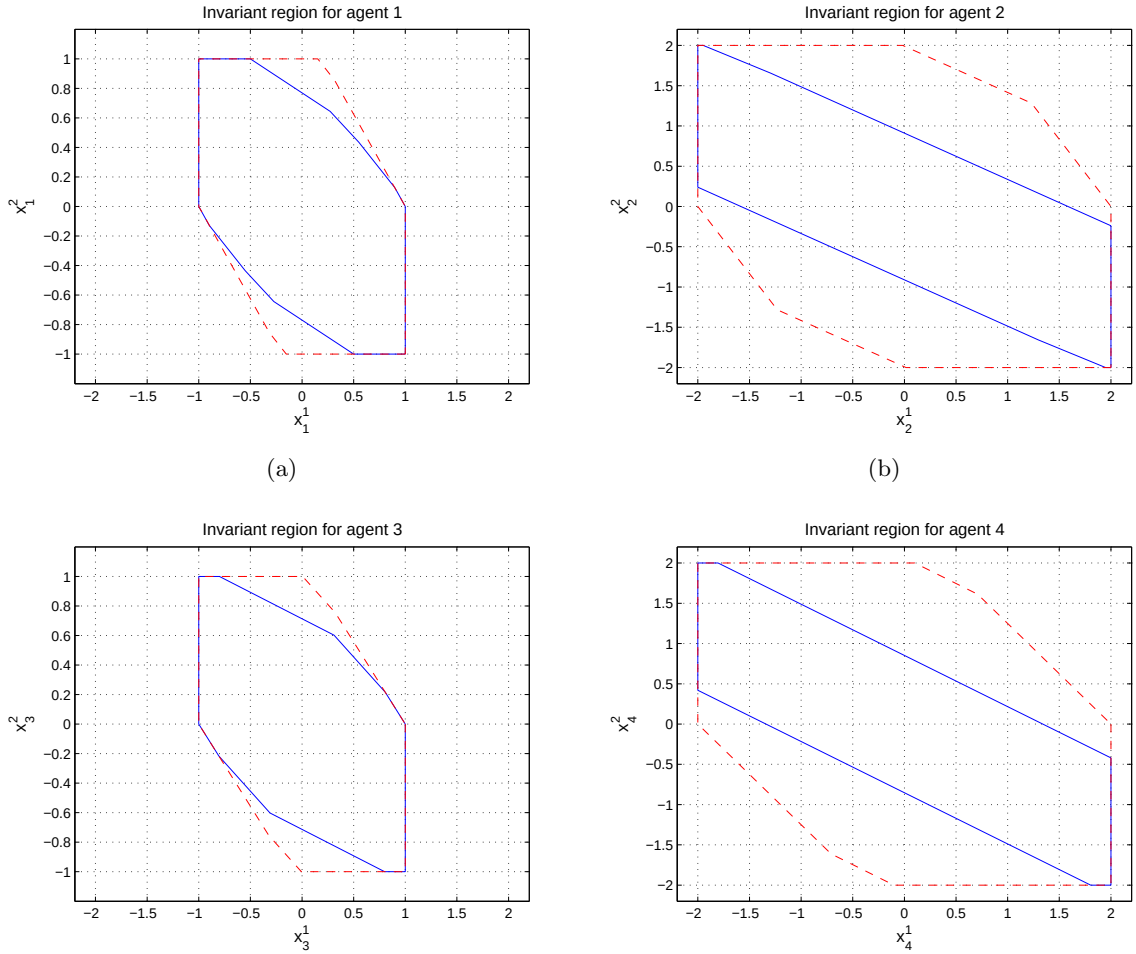
In general, the closed-loop stability properties are independent on how many proposals are evaluated or how this proposals are generated. This implies that the proposed controller scheme can be implemented using different proposal generation protocols. In this simulation, a communication protocol based on broadcast different from the one presented in Section 3 is used. At each sample time, each agent makes a single proposal optimizing its local cost function with respect to all the manipulated variables that affect him. All the proposals are compared (including U^s) and the one with the lower cost function is applied.

Figure 3.7 shows the closed-loop state trajectories of all the subsystems with the corresponding jointly invariant sets. The simulations presented were done for a prediction horizon $N = 12$, for the initial state

$$x_1(0) = \begin{bmatrix} -0.2311 \\ 0.9072 \end{bmatrix}, x_2(0) = \begin{bmatrix} -1.3558 \\ 0.9929 \end{bmatrix}, x_3(0) = \begin{bmatrix} -0.6533 \\ -0.2228 \end{bmatrix}, x_4(0) = \begin{bmatrix} -1.0419 \\ 1.1576 \end{bmatrix} \tag{3.66}$$

and an initial control vector $U^s(0)$ calculated as a feasible control vector for the centralized system for such initial state.

Figure 3.6 shows the proposal chosen at each time step. Numbers 1 to 4 indicate the agent that made the chosen proposal while 0 indicates that the shifted trajectory was chosen.



(a) (b)

(c) (d)

Figure 3.5: Jointly invariant set of each subsystem (solid lines) along with the corresponding projection of the centralized invariant set (dashed lines).

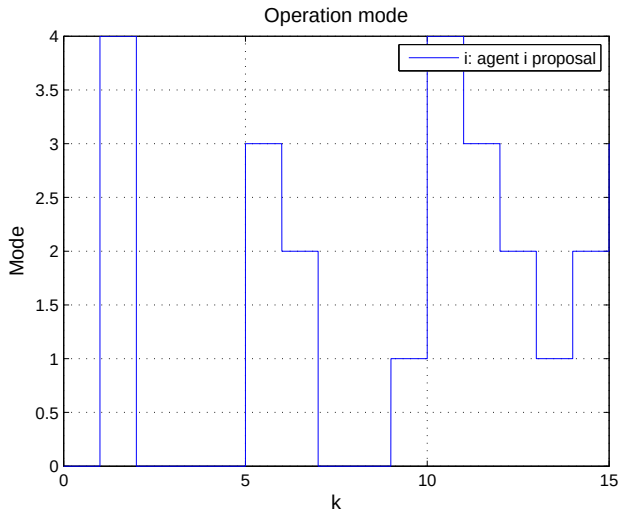


Figure 3.6: Proposal chosen at time k.

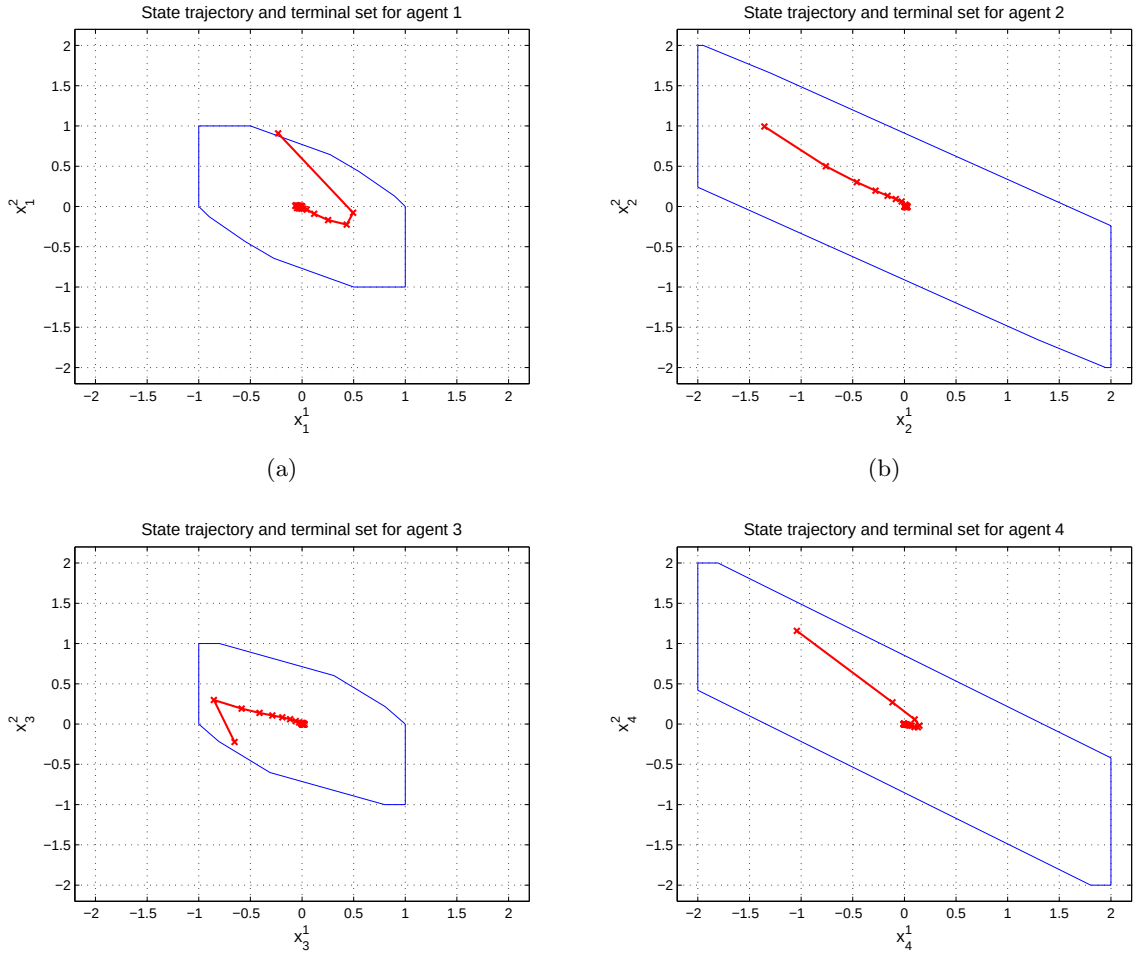
3.6 Application to a supply chain problem

In this section, we apply the proposed controller to a linear supply chain, which can be defined as the set of structures and processes used by an organization to provide a service or a good to a consumer. It is clear that the nodes of a supply chain may not have incentives to share other information about their models than their control actions. Supply chain flows usually present three interesting phenomena from the control point of view: oscillation, amplification and phase lag [92]. Due to material or informational delays production and inventories overshoot and undershoot the optimal levels. The magnitude of the fluctuations increase as they propagate from the customer to the factory, in what is commonly known as the bullwhip effect. For these reasons supply chains dynamics have been deeply analyzed and have been used as an application example in several distributed control papers [23, 54].

In this example, we consider a cascade of M_x firms. In particular, the discrete time equations that define the dynamics of firm i are given by:

$$s^i(t+1) = s^i(t) + u^{i-1}(t - d_{i-1,i}) - u^i(t) \tag{3.67}$$

The super-scripts $i-1$ and $i+1$ represent, respectively, the dynamics of the upstream and downstream nodes. Variable $s^i(t)$ is the stock level; that is, the number of items available for shipment downstream. The manipulated variable at each stage is $u^i(t)$ which stands for the number of items sent to the downstream node. This is a difference with respect to models in which there is one variable that stands for the order rate and another, which is usually modeled as a disturbance, that stands for the shipment itself. The information flows are



(a) (b)

(c) (d)

Figure 3.7: (a) Agent 1 state evolution. (b) Agent 2 state evolution. (c) Agent 3 state evolution. (d) Agent 4 state evolution.

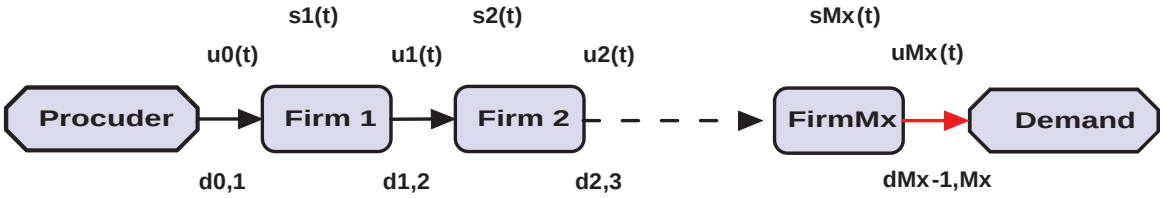


Figure 3.8: Linear supply chain.

assumed to have no time delays and the material flows have a delay modeled by $d_{i,j}$ which corresponds to the time taken by the shipments from node i to node j.

The only information shared by the agents is their inputs. In particular, the model of a node needs to keep track of the shipments made by its upstream node. This implies that a model of the form (3.1) can be obtained assigning a different subsystem to each firm i with the following state vector x_i

$$x_i(t) = \begin{bmatrix} s^i(t) \\ u^{i-1}(t-1) \\ u^{i-1}(t-2) \\ \vdots \\ u^{i-1}(t-d_{i-1,i}) \end{bmatrix}$$

Note that this model takes into account the different delays by augmenting the state vectors. The inputs are defined by the different shipments variables u^j.

In this model the first firm, with state $x_1(t)$ is the supplier which demands items directly to the factory by $u_0(t)$ which is modeled as a pure delay of value $d_{0,1}$. The last firm is the retailer which must satisfy the external demand $u_{M_x}(t)$ which is an external signal not controlled by the system. The control objective is to regulate the stock levels to a desired value $r^i(t)$. In addition, the last node of supply chain, the retailer, has to satisfy the external demand. To this end, we consider the following local cost function for each firm

$$J_i = \sum_{k=1}^{N} 2^i (r_k^i - s_k^i - \sum_{l=k-1}^{d_{i-1}} u_{k-l}^{i-1})^2$$

where N is the prediction horizon, the subindex k denotes the k-steps predicted value of a signal. No terminal cost function is considered. The cost penalizes the deviation of the sum of current stock and the items traveling from the upstream node from the desired reference. Note that if the controller ignores those units that have to arrive in the future, it would ask for more units than needed. The weights of the local cost grow with 2^i, that is, the closer a node is to the retailer the more important is. This way of weighting the error is natural since the most important goal of a supply chain is to satisfy the external demand.

The class of linear supply chains considered in this example is defined by the number of firms M_x and the delay parameters $d_{i,j}$. In the following tables we show the results of a set of simulations with three different supply chains of 5, 10 and 20 firms. We denote these scenarios as SUPPLY5, SUPPLY10 and SUPPLY20 respectively. The delay parameters of each supply chain have been randomly chosen with values between 2 and 5. The initial stock $s^i(0)$ was chosen randomly between 100 and 300. In all these simulations, we assume that the external demand $u_{M_x}(t)$ is null and that the objective of the controller is to regulate the stocks to their references. The references $r^i(t)$ were supposed to be constant and were chosen randomly between 180 and 280. The simulation times T_f were set respectively to 50, 100 and 200 sample times.

In order to study the effect of the number of proposals considered at each sampling time in the performance of the proposed DMPC scheme, we have applied several controllers which consider a different number of proposals N_{prop}. Given that the proposals are made randomly, each simulation was repeated 10 times. In addition, a centralized MPC controller has also been applied to the three scenarios as a reference of the performance that can be obtained with a centralized approach.

The tables show the cumulated cost mean $\bar{J}_{cum}$ and the corresponding standard deviation σ_J of each controller. The cumulated cost of each simulation was computed as:

$$J_{cum} = \sum_{t=0}^{T_f} \sum_{i=1}^{M_x} 2^i (r^i(t) - s^i(t))^2$$

In addition the tables show the mean number of sample times $\bar{t}_{ss}$ that an agent needs to have less than a 5% of error with respect to its reference, as well as the average number of sample times t_{ss} that the slowest agent needs to have less than a 5% of error with respect to its reference. These two entries provide additional information on the performance of each controller.

In general, the simulations show that increasing the number of proposals N_{prop} improves the performance of the proposed DMPC scheme. It can be seen that $\bar{J}_{cum}$ and σ_J are decreasing functions of the parameter N_{prop}. However, communications can be a scarce resource for some systems and it is important to find a trade-off between the number of communications and the performance. In our example it can be seen that a good trade-off happens when N_{prop} is around $5M_x$ communications, where M_x is the number of agents. This implies that each agent makes an average of 5 proposals to its neighbors.

Controller	$\bar{J}_{cum}$	σ_J	$\bar{t}_{ss}$	t_{ss}	N_{prop}
DMPC	2.36e+6	1.80e+6	25.57	30.70	1
DMPC	9.39e+5	2.99e+6	15.20	18.30	3
DMPC	6.64e+5	3.05e+5	11.25	13.80	5
DMPC	5.53e+5	1.40e+5	9.85	12.20	7
DMPC	5.39e+5	1.69e+5	9.05	11.30	10
DMPC	4.34e+5	8.30e+4	8.25	10.50	15
DMPC	3.88e+5	1.25e+4	7.70	9.50	20
DMPC	3.86e+5	1.24e+4	7.65	9.30	30
MPC	3.71e+5	-	7.50	9.00	-

Table 3.1: Simulation results for SUPPLY5

Controller	$\bar{J}_{cum}$	σ_J	$\bar{t}_{ss}$	t_{ss}	N_{prop}
DMPC	8.50e+7	1.95e+7	58.32	93.50	1
DMPC	4.57e+7	1.20e+7	29.48	46.30	3
DMPC	2.61e+7	3.28e+6	21.78	34.30	5
DMPC	2.62e+7	4.23e+6	20.34	29.50	7
DMPC	2.06e+7	2.98e+6	16.18	24.20	10
DMPC	1.71e+7	1.98e+6	13.81	21.30	15
DMPC	1.70e+7	2.00e+6	13.68	21.50	20
DMPC	1.63e+7	1.25e+6	12.82	20.50	30
DMPC	1.53e+7	9.56e+5	12.91	20.50	50
DMPC	1.52e+7	7.07e+5	12.36	20.10	70
DMPC	1.52e+7	6.51e+5	12.07	20.10	100
MPC	1.45e+7	-	13.00	20.00	-

Table 3.2: Simulation results for SUPPLY10

Controller	$\bar{J}_{cum}$	σ_J	$\bar{t}_{ss}$	t_{ss}	N_{prop}
DMPC	5.84e+11	1.24e+11	162.49	199.80	1
DMPC	2.66e+11	6.02e+10	132.96	187.00	3
DMPC	1.46e+11	2.83e+10	93.34	137.50	5
DMPC	1.30e+11	1.48e+10	76.09	112.90	7
DMPC	9.85e+10	1.73e+10	60.21	88.90	10
DMPC	8.23e+10	1.11e+10	49.48	73.40	15
DMPC	6.19e+10	7.09e+9	42.24	61.70	20
DMPC	5.55e+10	4.52e+9	39.02	56.40	30
DMPC	5.24e+10	3.01e+9	32.38	46.70	50
DMPC	5.07e+10	1.38e+9	31.15	44.30	70
DMPC	5.01e+10	9.34e+8	30.36	42.90	100
DMPC	4.98e+10	8.90e+8	29.91	42.10	125
DMPC	4.97e+10	5.79e+8	29.65	41.80	150
DMPC	4.98e+10	4.03e+8	29.23	41.90	175
DMPC	4.95e+10	6.09e+8	28.31	41.10	200
MPC	3.84e+10	-	19.53	26.00	-

Table 3.3: Simulation results for SUPPLY20

3.7 Application to control of irrigation canals

In this chapter we apply the proposed controller to a model of a section of the "postrasvase Tajo-Segura" in the south-east of Spain. The 'postrasvase Tajo-Segura' is a set of canals which distribute water coming from the Tajo river in the basin of the Segura river. This water is mainly used for irrigation (78%), although a 22% of it is drinking water. The selected section is a Y-shape canals (see figure 3.9), a main canal that splits into two canals with a gate placed at the input of each one of them: Canal de la Pedrera, with a total length of 6,680 kilometres, and Canal de Cartagena, with a lenght of 17,444 kilometres.

The total length of the canals is approximately of 24 kilometres. At the end of Canal de Cartagena there is a reservoir with limited capacity.

The main elements in the canals are the main gates, which regulate the level of water along the canals, and the off-take gates, where the farmers take water from the canals for irrigation. There are 7 main gates and 17 off-take gates in the section studied.

Figure 3.9 and table 3.4 show a scheme of the location of the gates, the off-take gates and the milestones where they are located.

Code	Type	P/G	Description	Km
			Canal del Campo de Cartagena	
			Starting Campo de Cartagena canal	0,000
CCMICAR-01	Gate	G	Initial Gate	0,200
MICAR-01	Off-take	G	Off-take 5 - Fuensanta and Estafeta	1,170
MICAR-02	Off-take	G	Off-take 5' - Palacete	2,540
MICAR-03	Off-take	P	Off-take 6 - Santo Domingo	2,840
CCMICAR-04	Gate		Gate Canal Pedrera	4,485
MICAR-04	Off-take	P	Off-take 7 - Campo Salinas	5,970
MICAR-05	Off-take	G	Off-take 8 - San Miguel	6,550
MICAR-06	Off-take	G	Off-take 9 - Las Caadas	8,050
MICAR-07	Off-take	G	Off-take 10 - San Miguel	9,390
MICAR-08	Off-take	P	Off-take 11 - Campo Salinas	9,590
CCMICAR-05	Gate		Gate Tunel San Miguel	10,480
MICAR-09	Off-take	G	Off-take 12 - San Miguel	12,630
MICAR-10	Off-take	P	Off-take 13 - Campo Salinas	12,780
CCMICAR-06	Gate		Gate La Rambla La Fayona (start)	14,433
CCMICAR-07	Gate		Gate La Rambla La Fayona (end)	14,579
MICAR-11	Off-take	P	Off take 14 - Villamartin	16,540
CCMICAR-08	Gate		Gate Caada La Estacada	17,444
			Canal de la Pedrera	
CCMIPED-01	Gate		Starting La Pedrera canal	0,000
MIPED-01	Off-take	G	Off-take 1P - S. Domingo	0,770
MIPED-02	Off-take	G	Off-take 2P - S. Domingo y Mengoloma	3,740
MIPED-03	Off-take	P	Off-take 3P - S. Domingo	4,260
MIPED-04	Off-take	G	Off-take Riegos Levante 1	5,260
MIPED-05	Off-take	G	Off-take 4P - Santo Domingo	6,440
MIPED-06	Off-take	G	Off-take Riegos Levante 2 y 3	6,680

Table 3.4: Data of irrigation canal Cartagena-La Pedrera

Figure 3.9: Scheme of the canal.

The dynamics of water flowing in irrigation open canals can be obtained by applying the Saint Venant equations [16, 17], which are nonlinear partial differential equations.

The irrigation canals considered are divided into several sections separated by gates; the controlled variables are the downstream water levels, $h_i(t) \in \mathbb{R}^+(m)$ and the manipulated variables are the check point to gates, $u_i(t) \in \mathbb{R}^+(m)$.

Each canal reach has an inflow from an upstream canal reach, $Q_{in,i} \in \mathbb{R}^+(m^3/s)$, and an outflow to a downstream canal reach, $Q_{o,i} \in \mathbb{R}^+(m^3/s)$. Also, other flows are considered as perturbation variables. In particular, $q_{in,i} \in \mathbb{R}^+(m^3/s)$ models the flows due to rainfall, failures in upstream gate and other unknown disturbances and $q_{o,i} \in \mathbb{R}^+(m^3/s)$ models the known offtake outflows from farmers, considered as measurable perturbations.

The discrete model that has been considered using the previous variables is:

$$A_i(h_i(k+1) - h_i(k)) = T_d(Q_{in,i}(k - t_d) + q_{in,i}(k) - Q_{o,i}(k) - q_{o,i}(k)) \tag{3.68}$$

where $T_d(s)$ is the sampling time, A_i is the surface of the reach and t_d the delay of the input Q_{in} (the level is measured downstream).

The discharge through a submerged flow gate $Q_o(t)$ is usually determined as[16]:

$$Q_o(t) = C_d L\sqrt{2g}u(t)\sqrt{h_{up}(t) - h_{dn}(t)}, \tag{3.69}$$

where C_d is the gate discharge coefficient, L is the gate width, $u(t)$ the gate opening and $h_{up}(t), h_{dn}(t)$ are the upstream and downstream water levels, respectively.

The main target is to manage the water in the canals in order to guarantee the flows requested by users. Another objective to be considered is the minimization of the leaks and evaporation (which can be obtained as function of the levels) and also to minimize maintenance costs (the maintenance of concrete blocks and junctions is better if they are submerged, so high levels are preferred for that purpose). For these purposes, it is necessary to maintain the level of the canal over the off-take gate when flow is requested. The controlled variables are the upstream levels beside the gates, which have to satisfy maximum and minimum level constraints. The minimum level is determined by the demand of irrigation of surrounded lands; it must be guaranteed that off-take points are submerged. The maximum level is determined to provide regulation capabilities to the system in order to avoid floods. The manipulated variables are the flow at the head of the canal and the position of the gates, which are are also constrained. The flow at the head is limited by the total amount of available water and the gates have maximum and minimum openings.

The actual reference in levels are sent to the DMPC in the low level. For this controller, the sample time has been considered 1 minute and the horizon, N, has been set to 5. The cost function has been designed to include the water traveling from the upstream gates, that is, the states which correspond to the traveling water are also weighted. It is necessary to consider the water coming from the upstream gate as a part of the level under consideration to be able to compensate the effect of the delay in the controller. The weights of the local costs in the canals grow with 2^i, that is, the farther a node is from the beginning , the more important is. This way of weighting the error facilitates a faster flow of water towards the last canals. Finally, the matrix that weights the control effort R_i has been set to zero for simplicity.

In figure 3.10 it can be seen a simulation in which all the reaches begin with a water level of 3 meters. At sampling time $k = 0$, the reference is set to $3.15m$ to all the reaches. After a whole day ($k = 1440$) there is another change of reference for all the reaches to $3.6m$. These changes are originated in the upper control level as a function of the risk mitigation policy. It can be seen how the level in the reaches follow the reference even in the presence of disturbances (farmers take water, rains...). It is important to remark that these results have been obtained with an average number of 5 communications per agent and sampling time.

In other words, each agent makes up to 5 proposals in a minute in order to get a cooperative solution with the rest of the agents.

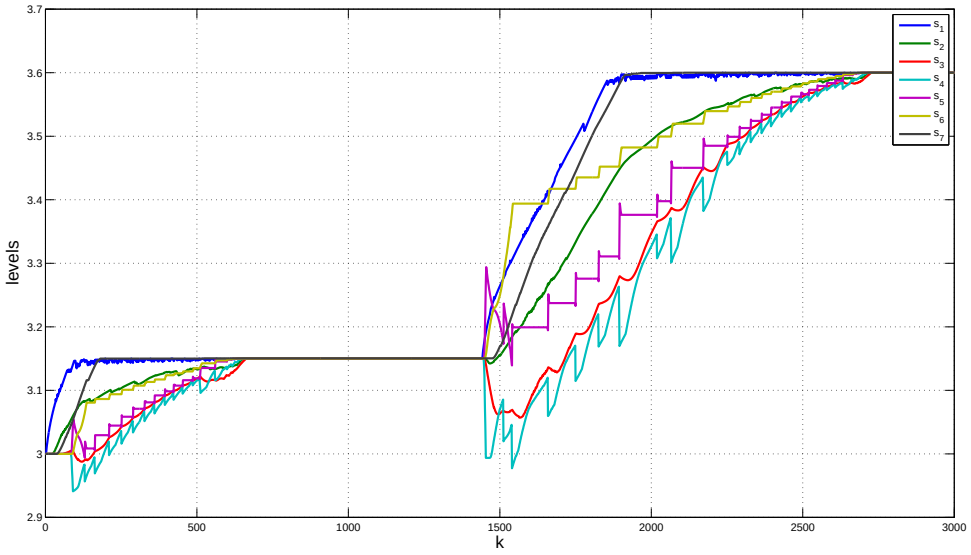


Figure 3.10: Water level evolution.

3.8 Conclusions

In this chapter we have presented a novel distributed MPC algorithm based on negotiation for a class distributed linear systems coupled through the inputs. We assume that each agent has access only to the model and the state of one of the subsystems and that the agents must negotiate in order to reach a cooperative solution. The proposed algorithm has low communication and computational burdens and provides a feasible solution to the centralized problem. In addition, we provide sufficient conditions that guarantee practical stability of the closed-loop system as well as an optimization based procedure to design the controller so that these conditions are satisfied.

Chapter 4

Distributed Receding Horizon Kalman Filter

The most common approach to estimate the state of stochastic systems is the Kalman filter [36], developed in 1960 and named after his discoverer. The Kalman filter is the optimal state estimator for unconstrained linear systems subject to gaussian state and output noise. It is not possible to apply directly the centralized Kalman filter to a multiagent problem unless there is a node in the network that receives all the information. For example in [87], it can be seen how a central agent gathers the information from the moving devices and then distributes the position estimation back to them. An alternative is to calculate a decentralized version of the Kalman filter that takes into account the communications restrictions [31, 87].

In this thesis we follow a different approach to solve the estimation problem in a distributed manner. First, the Kalman filter is posed as a dynamic programming problem [18]. Then, the resulting optimization problem is distributed among the agents by means of dual decomposition. This idea has been successfully applied to distributed control in [82] and [28]. Given that the observation problem is the dual of the control problem, it is natural to apply and enhance the techniques presented in these papers to deal with the state estimation problem. Moreover, some of the results that will be shown in this chapter have direct application to distributed control based on dual decomposition.

In this context, the application of state estimation schemes to problems in which the state represents the position of an object is very attractive [4], [31]. The localization of moving entities, such as robots or people, is important for many applications. Military applications in which the goal is to track a target that moves in a distributed sensor environment are typical examples. Other examples in which these techniques play an important role would

be smart homes [49], in which it is basic to know where the inhabitants of the house are in order to control the heating and the lights and traffic and speed control. For this reason, they constitute good applications for the distributed state estimation algorithm that we present.

The outline of the chapter is as follows. In section I the problem is formulated. Section II explains how dual decomposition can be used to distribute the problem among the agents involved. In section III the techniques presented in the previous sections are applied in simulation examples. Finally, conclusions and future work are presented.

Part of this chapter has been published in [50].

4.1 Problem formulation

In this section we present a moving horizon estimation strategy that solves approximately the Kalman filter. Let us consider the following uncertain distributed linear system

$$\begin{aligned} x_i(\tau+1) &= A_{ii}x_i(\tau) + w_i(\tau) \\ y_i(\tau) &= \sum_{j=1}^{J} C_{ij}x_j(\tau) + v_i(\tau) \end{aligned} \tag{4.1}$$

where $x_i(\tau) \in \mathbb{R}^{n_i}$, $y_i(\tau) \in \mathbb{R}^{q_i}$, $w_i(\tau) \in \mathbb{R}^{n_i}$ and $v_i(\tau) \in \mathbb{R}^{q_i}$ are the state, measurable output, state noise and measurement noises of the i-th subbsytem respectively. The state and measurement noises are characterized by a normal distribution with zero mean and variances Q_i and R_i respectively; that is, $w_i(\tau)$ is a $N(0, Q_i)$ and $v_i(\tau)$ is a $N(0, R_i)$. From a centralized point of view the system can be described with the following model

$$\begin{aligned} x(\tau+1) &= Ax(\tau) + w(\tau) \\ y(\tau) &= Cx(\tau) + v(\tau) \end{aligned} \tag{4.2}$$

where

$$\begin{aligned} x(\tau) &= [x_1(\tau)\ x_2(\tau)\ \dots\ x_J(\tau)]^T \in \mathbb{R}^n \\ y(\tau) &= [y_1(\tau)\ y_2(\tau)\ \dots\ y_J(\tau)]^T \in \mathbb{R}^q \\ w(\tau) &= [w_1(\tau)\ w_2(\tau)\ \dots\ w_J(\tau)]^T \in \mathbb{R}^n \\ v(\tau) &= [v_1(\tau)\ v_2(\tau)\ \dots\ v_J(\tau)]^T \in \mathbb{R}^q \\ n &= \sum_i n_i,\ q = \sum_i q_i. \end{aligned}$$

Note that $w(\tau)$ is a $N(0, Q)$ with $Q = diag(Q_i)$ for $i = 1, \ldots, J$ and $v(\tau)$ is a $N(0, R)$ with $R = diag(R_i)$ for $i = 1, \ldots, J$.

From the point of view of probability theory, a state estimator attempts to reconstruct the a posteriori distribution $p(\hat{x}(\tau)|Y(0:\tau))$, which is the probability that the state of the system is $\hat{x}(\tau)$ given measurements $Y(0:\tau) = \{y(0), \ldots, y(\tau)\}$. It is also possible to calculate the joint probability for a trajectory of state values, for example $p(\hat{X}(0:\tau)|Y(0:\tau))$. It is clear that if the distribution can be calculated, then it is possible to obtain an estimate that maximizes it, that is,

$$\hat{X}^*(0:\tau) = arg \max_{\hat{X}(0:\tau)} p(\hat{X}(0:\tau)|Y(0:\tau)). \tag{4.3}$$

Note that the number of optimization variables involved in the estimation optimization problem grows with each new sample that has to be estimated. In order to bound the computational burden it is possible to estimate the state trajectory inside a window of size N (see for example [30]). In this case, equation (4.3) becomes

$$\hat{X}^*(\tau - N, \tau) = arg \max_{\hat{X}(\tau-N,\tau)} p(\hat{X}(\tau - N, \tau)|Y(0:\tau)). \tag{4.4}$$

We will also use this approximation in the approach presented in this chapter to build a distributed version of the Kalman filter. In this case, the problem of obtaining an estimate that maximizes the probability density function can be reduced to a dynamical programming problem. See [4] or [85] to obtain more details. In particular, the maximization of $p(\hat{X}(\tau-N : \tau)|Y(0:\tau))$ can be solved in a recursive fashion with the introduction of the following auxiliary function

$$I(\hat{x}(\tau)) = \max_{\hat{X}(\tau-N:\tau-1)} p(\hat{X}(\tau - N : \tau - 1), \hat{x}(\tau)|Y(0:\tau)), \tag{4.5}$$

which can be interpreted as the probability of the most probable trajectory that reaches $\hat{x}(\tau)$. We assume that $p(\hat{x}(\tau)|\hat{X}(\tau - N : \tau - 1)) = p(\hat{x}(\tau)|\hat{x}(\tau - 1))$, that is, the state in time τ depends only on the state in time $\tau - 1$. This assumption is known as the Markov assumption and, together with Bayes' Theorem, allows to rewrite the equation (4.5) as

$$I(\hat{x}(\tau)) = \max_{\hat{x}(\tau-1)} \frac{p(y(\tau)|\hat{x}(\tau))p(\hat{x}(\tau)|\hat{x}(\tau-1))}{p(y(\tau)|Y(0,\tau-1))} I(\hat{x}(\tau - 1)). \tag{4.6}$$

Note that it is possible to maximize individually the probability of all the state transitions from from $\hat{x}(k)$ to $\hat{x}(k+1)$ for all $k \in [\tau - N, \tau - 1]$ due to the Markov assumption. Note too that the term $p(y(\tau)|Y(0:\tau-1))$ can be discarded since it does not depend on the optimization variable. In addition, note that if we take into account the system dynamics given by (4.2), it is possible to calculate explicitly $p(y(\tau)|\hat{x}(\tau))$ and $p(\hat{x}(\tau+1)|\hat{x}(\tau))$. Simply, taking into account that

$$\begin{aligned} w(\tau) &= x(\tau+1) - Ax(\tau) \\ v(\tau) &= y(\tau) - Cx(\tau). \end{aligned}$$

It is clear that $p(y(\tau)|x(\tau)) = p(v(\tau))$ and $p(x(\tau+1)|x(\tau)) = p(w(\tau))$. Given that $w(\tau)$ and $v(\tau)$ are assumed to be normal random variables we can obtain their corresponding probability density functions as

$$\begin{aligned} p(\hat{x}(i+1)|\hat{x}(i)) = p(w(\tau)) &= \frac{1}{(2\pi)^{n/2}\sqrt{|Q^{-1}|}} e^{-\frac{1}{2}w(\tau)^T Q^{-1} w(\tau)} \\ p(y(i)|\hat{x}(i)) = p(v(\tau)) &= \frac{1}{(2\pi)^{q/2}\sqrt{|R^{-1}|}} e^{-\frac{1}{2}v(\tau)^T R^{-1} v(\tau)}. \end{aligned} \tag{4.7}$$

The recursive equation (4.6) can be transformed into a dynamic programming problem applying the logarithm operation to the both sides or the equality, that is,

$$\log(I(\hat{x}(\tau))) = \max_{\hat{x}(\tau-1)} (\log p(y(\tau)|\hat{x}(\tau)) + \log p(\hat{x}(\tau)|\hat{x}(\tau-1)) + \log(I(\hat{x}(\tau-1)))).$$

Therefore, the most probable trajectory of states $\hat{X}^*(\tau - N, \tau)$ can be calculated in the following way:

$$\hat{X}^*(\tau - N, \tau) = arg \min_{\hat{X}(\tau-N,\tau)} (- \sum_{k=\tau-N+1}^{\tau} \log p(y(k)|\hat{x}(k)) + \log p(\hat{x}(k)|\hat{x}(k-1)) + \Phi(\hat{x}(\tau - N))), \tag{4.8}$$

where $\Phi(\hat{x}(\tau - N)) = \log(I(\hat{x}(\tau - N)))$ is a term that weights the uncertainty of the first state estimated in the window. Note that we have changed the maximization to a minimization by changing the sign of all the terms of the objective function.

Substituting (4.7) into equation (4.8), the moving horizon estimation problem can be posed as a quadratic programming optimization problem. In addition, the logarithm function

allows us to ignore the terms $1/((2\pi)^{n/2}\sqrt{|Q^{-1}|})$ and $1/((2\pi)^{q/2}\sqrt{|R^{-1}|})$ in the cost function, which become simply constants (assuming that R and Q are constant matrices). In order to simplify the introduction of the quadratic programming problem, we define next the following quadratic function:

$$\begin{aligned} V^N(\hat{X}(\tau-N:\tau)) = &\sum_{k=\tau-N+1}^{\tau} \tfrac{1}{2}(y(k)-C\hat{x}(k))^T R^{-1}(y(k)-C\hat{x}(k)) \\ + &\sum_{k=\tau-N}^{\tau-1} \tfrac{1}{2}(\hat{x}(k+1)-A\hat{x}(k))^T Q^{-1}(\hat{x}(k+1)-A\hat{x}(k)) + \Phi(\hat{x}(\tau-N)), \end{aligned} \tag{4.9}$$

Remark 11 *Note that equation (4.9) can expressed as the sum of a stage cost for each estimate but the last one, which value is calculated through the terminal cost. According to this,*

$$V^N(\hat{X}(\tau-N:\tau)) = \sum_{k=\tau-N+1}^{\tau} l(\hat{x}(k)) + \Phi(\hat{x}(\tau-N)). \tag{4.10}$$

Remark 12 *The terminal cost in equation (4.9) is commonly referred as the arrival cost. This term summarizes the information not considered in the horizon at time τ. In the case considered, that is, linear model and gaussian noises, this term would simply become $\Phi(\hat{x}(\tau-N)) = \|\hat{x}(\tau-N)-m\|^2_{P^{-1}(\tau-N)}$ [85], where $P^{-1}(\tau-N)$ is the inverse of the covariance matrix of the estimation error and m is the mean of $x(\tau-N)$. Nevertheless, it is not practical in a distributed dynamic programming problem to keep track of $P^{-1}(\tau-N)$ and approximations are needed. One possible choice is to use the steady state covariance matrix to weight the estimation at the beginning of the window. In this thesis the problem will be relaxed assuming that $x(\tau-N)$ takes the value calculated in its latest estimation $\hat{x}(\tau-N)$. This assumption works well as long as the previous estimates are correctly estimated. Actually, in the case that the trajectory of estimated states out of the estimation window were all exact (which, of course, is highly improbable) then this approximation would become just an application of Bellman's principle of optimality [10].*

The optimal estimation for the trajectory of states $\hat{X}^*(0:\tau) = \{\hat{x}^*(0),\ldots,\hat{x}^*(\tau)\}$ is obtained solving the following minimization problem

$$\hat{X}^*(0:\tau) = arg\min_{\hat{X}(0:\tau)} V^\tau(\hat{X}(0:\tau)) \tag{4.11}$$

subject to (4.2) and taking $\Phi(x(0)) = \|x(0) - m(0)\|^2_{P^{-1}(0)}$. This problem is equivalent to the Kalman filter [18] but it has a major drawback: the computational burden of (4.11) grows with τ as more measurements become available. We use an approximate moving horizon estimation approach to fix the computational cost. The estimation we make is $\hat{X}(\tau - N : \tau) = \{\hat{x}(\tau - N), \ldots, \hat{x}(\tau)\}$ and can be calculated solving the following QP problem:

$$\hat{X}^*(\tau - N : \tau) = arg \min_{\hat{X}(\tau - N:\tau)} V^N(\hat{X}(\tau - N : \tau)) \tag{4.12}$$

subject to (4.2) and $x(\tau - N) = \hat{x}(\tau - N)$.

Remark 13 *Note that the state equation in (4.2) can be used to determine the noise trajectory once the state trajectory has been calculated. This relationship can be used in the opposite way so that the QP problem can also be solved minimizing with respect the noise trajectory* $w(\tau - N), .., w(\tau - 1)$. *Taking into account the duality between the control and estimation problems, a possible interpretation for the minimization alternative is that the term* $w_i(\tau)$ *is used to* control *the estimation.*

4.2 Dual decomposition

The goal of this chapter is to distribute the estimation problem between all the agents present in the system. Under certain assumptions, in [28] dual decomposition was used to distribute the optimization problem corresponding to a MPC controller between several agents. As the problem of estimation is the dual of the control problem, and we have reduced the estimation to the optimization of a cost function, the same methodology will be applied.

It can be seen in equation (4.1) that the outputs of the subsystems are coupled through the states. The coupling term represents the effect of the rest of the subsystems in the measurements of agent i. We will define $d_i(\tau) = \sum_{i \neq j} C_{ij} x_j(\tau)$ to denote this effect. The subsystem model can be rewritten as

$$\begin{aligned} x_i(\tau + 1) &= A_{ii} x_i(\tau) + w_i(\tau) \\ y_i(\tau) &= C_{ii} x_i(\tau) - d_i(\tau) + v_i(\tau) \end{aligned} \tag{4.13}$$

subject to the constraint $d_i(\tau) = -\sum_{i \neq j} C_{ij} x_j(\tau)$.

Dual decomposition can be used to distribute the centralized problem (4.12) between the agents. The introduction of Lagrange multipliers p_i in the cost function allows the distribution of the cost function (4.9). First, we define the Lagrange extended cost function as

$$\begin{aligned} &V^{N,p}(\hat{X}(\tau-N:\tau), D(\tau-N:\tau), P(\tau-N:\tau)) = \\ &\sum_{i=1}^{J}\{\sum_{k=\tau-N}^{\tau-1} \|\hat{x}_i(k+1) - A\hat{x}_i(k)\|^2_{Q_i^{-1}(k)} \\ &+ \sum_{k=\tau-N}^{\tau} \|-C_{ii}\hat{x}_i(k) + y_i(k) + d_i(k)\|^2_{R_i^{-1}(k)} \\ &+ \sum_{k=\tau-N}^{\tau} p_i^T(k)(d_i(k) + \textstyle\sum_{i\neq j} C_{ij}\hat{x}_j(k))\} \end{aligned} \tag{4.14}$$

where $p_i(\tau) \in \mathbb{R}^{q_i}$ is the lagrange multiplier corresponding to the constraint induced by $d_i(\tau) \in \mathbb{R}^{q_i}$, which is now a free variable. Their corresponding centralized vectors are respectively $p(\tau) = [p_1(\tau)\ p_2(\tau)\ \dots\ p_J(\tau)]^T \in \mathbb{R}^q$ and $d(\tau) = [d_1(\tau)\ d_2(\tau)\ \dots\ d_J(\tau)]^T \in \mathbb{R}^q$. Finally, we denote the sequences of these vectors in time as $P(\tau-N:\tau) = \{p(\tau-N), \dots, p(\tau)\}$ and $D(\tau-N:\tau) = \{d(\tau-N), \dots, d(\tau)\}$.

If we take $Q_i^{-1}(\tau) = 0$ in 4.14 we can reduce the two summations to one. Then, if we rearrange the lagrangian multipliers it is possible to rewrite the extended cost function as:

$$\begin{aligned} &V^{N,p}(\hat{X}(\tau-N:\tau), D(\tau-N:\tau), P(\tau-N:\tau)) = \\ &\sum_{i=1}^{J}\sum_{k=\tau-N}^{\tau} [\|\hat{x}_i(k+1) - A\hat{x}_i(k)\|^2_{Q_i^{-1}(k)} \\ &+ \|-C_{ii}\hat{x}_i(k) + y_i(k) + d_i(k)\|^2_{R_i^{-1}(k)} \\ &+ p_i^T(k)d_i(k) + \hat{x}_i(k)^T \textstyle\sum_{i\neq j} C_{ji}^T p_j(k)] \\ &= \sum_{i=1}^{J} V_i^{N,p}(\hat{X}_i(\tau-N:\tau), D_i(\tau-N:\tau), P(\tau-N:\tau)) \end{aligned}$$

The quadratic problem can be distributed among the agents because the local extended cost functions $V_i^{N,p}(\hat{X}_i(\tau-N:\tau), D_i(\tau-N:\tau), P(\tau-N:\tau))$ are decoupled. From a centralized point of view the problem that is solved at each time sample is

$$\max_{P(\tau-N:\tau)} \sum_{i=1}^{J} \min_{\substack{\hat{X}_i(\tau-N:\tau),\\ D_i(\tau-N:\tau)}} V_i^{N,p}\begin{pmatrix} \hat{X}_i(\tau-N:\tau),\\ D_i(\tau-N:\tau),\\ P(\tau-N:\tau)\end{pmatrix}$$

Remark 14 *If we define the stage cost at the time sample k as*

$$l_i(\hat{x}_i(k), d_i(k)) = \|-C_{ii}\hat{x}_i(k) + y_i(k) + d_i(k)\|^2_{R_i^{-1}(k)} + \|\hat{x}_i(k+1) - A\hat{x}_i(k)\|^2_{Q_i^{-1}(k)}$$

Then, the extended local cost function can be posed as

$$V_i^{N,p}(\hat{X}_i(\tau-N:\tau), P(\tau-N:\tau)) = \sum_{k=\tau-N}^{\tau} [l_i(\hat{x}_i(k), d_i(k)) + p_i^T(k) d_i(k) + \hat{x}_i(k)^T \textstyle\sum_{i\neq j} C_{ji}^T p_j(k)]$$

The local stage cost can also be extended to include the terms due to the lagrangian prices $l_i^p(\hat{x}_i(k), d_i(k), P(k)) = l_i(\hat{x}_i(k), d_i(k)) + p_i^T(k) d_i(k) + \hat{x}_i(k)^T \sum_{i\neq j} C_{ji}^T p_j(k)$*, which allows to write the extended local cost function as:*

$$V_i^{N,p}(\tau) = \sum_{k=\tau-N}^{\tau} l_i^p(\hat{x}_i(k), d_i(k), P(k)) \tag{4.15}$$

Remark 15 *After the introduction of dual variables, and assuming that the prices of the neighbors are given, it is possible to interpret the distributed optimization procedure in economic terms. Each agent behavior can be represented as a two player game. The first player objective is to minimize the price-extended stage cost*

$$\sum_{k=\tau-N}^{\tau} l_i^p(\hat{x}_i(k), d_i(k), P(k)),$$

which is composed of three elements that are interpretable as

$$l_i^p(\hat{x}_i(k), d_i(k), P(k)) = \underbrace{l_i(\hat{x}_i(k), d_i(k))}_{\text{local cost}} + \overbrace{p_i^T(k) d_i(k)}^{\text{neighbor help cost}} + \underbrace{\hat{x}_i(k)^T \sum_{i\neq j} C_{ji}^T p_j(k)}_{\text{incomes due to required help}} .$$

The second player chooses the prices $p_i(\tau-N), \ldots, p_i(\tau)$ *to maximize*

$$p_i^T(k)(d_i(k) + \sum_{i\neq j} C_{ij}\hat{x}_j(k)).$$

This game is repeated iteratively. First, an estimate is calculated according to the given prices. Then, the prices are updated and the cycle starts again. As a result of the repeated interaction

of both players in each node the prices evolve until a maximum is reached. The consequence of this standard Lagrangian optimization procedure is that the minimum for the cost function (4.12) is attained and the constraints are satisfied when the price gradient is zero.

The algorithm that is followed by the agents in the system can be summarized as:

- Step 1: Each agent i estimates his own current state trajectory $\{\hat{x}_i(\tau - N), \hat{x}_i(\tau - N + 1), .., \hat{x}_i(\tau)\}$ solving the optimization problem given in (4.15) for a set of given prices p_i $i = 0, \ldots, J$.
- Step 2: Once the state trajectory has been calculated then the prices of agent i are updated by a gradient step as follows.

$$p_i^{k+1}(\tau) = p_i^k(\tau) + \gamma_i^k[d_i(\tau) + \sum_{i \neq j} C_{ij}\hat{x}_j(k)] \tag{4.16}$$

  Convergence of such gradient algorithms has been proved under different type of assumptions on the step size sequence γ_i^k. See for example [91]. Note that in order to update the prices the agents must communicate.

- Step 3: If the precision obtained with the estimation is enough then there is no need to continue iterating. In the next section precise conditions are given. If enough precision is not attained and the number of iterations K exceeds a given threshold *maxiter*, then the algorithm also stops. In other case then the process is repeated from step 1 for $K = K + 1$.

4.2.1 Coordination alternatives for the price update

It can be seen that the calculation of the estimate $\hat{x}_i(t)$ for $t = \tau - N, \ldots, \tau$ is completely decentralized once that prices are given. Therefore it is mandatory for an agent to keep the track of its neighbor prices. Nevertheless, in order to update the prices, coordination among the agents is necessary. The agents send their estimates $\hat{x}_i(\tau)$ to their neighbors so that equation (4.16) can be applied. For some systems it could be desirable not to share the state information with their neighbors. To avoid the exchange of the state estimates we propose two alternatives:

- Decentralized approach: The need for the shared information comes from term $\sum_{i \neq j} C_{ij}\hat{x}_j(k)$ in equation equation (4.16). According to the dynamics of the subsystems $\sum_{j \neq i} C_{ij}x_i(\tau) =$

$y_i(\tau) - C_{ii}x_i(\tau) - v_i(\tau)$, and thus it could be approximated by $\sum_{j\neq i} C_{ij}\hat{x}_i(\tau) \approx y_i(\tau) - C_{ii}\hat{x}_i(\tau)$.

- Market approach: This alternative consists on changing the way in which prices are updated. To understand better this approach it is convenient to use the behavior model that represents each agent as a two player game. Then, it is possible to think on the centralized problem as a game with $2J$ players. The objective of the first player in each node is to minimize his own cost according to the given prices. However, the second player in each node bargains with the the rest of the second players to maximize (4.15) with respect to the prices. The second players can be seen as market makers that fix the prices of the help services that the agents provide each other according to the offer and demand of such services. To do so, a gradient optimization of the cost function (4.15) is implemented. Each update is based on the addition of contributions of the different agents. The contribution of agent i is

$$\nabla p_i^k(\tau) = \begin{bmatrix} \hat{x}_i(k)^T C_{1i}^T p_1(k) \\ \vdots \\ \hat{x}_i(k)^T C_{i-1,i}^T p_{i-1}(k) \\ d_i(k) \\ \hat{x}_i(k)^T C_{i+1,i}^T p_{i+1}(k) \\ \vdots \\ \hat{x}_i(k)^T C_{Ji}^T p_J(k) \end{bmatrix}$$

Theorem 6 *The price update mechanism defined in the market approach provides the same results than the one presented in equation (4.16).*

Proof:

It is straight forward to check that both methods provide the same centralized price vector. It is enough to sum the contribution $\nabla p_i^k(\tau)$ for all i

$$p^{k+1}(\tau) = p^k(\tau) + \gamma \sum_i \nabla p_i^k(\tau).$$

Then it can be seen that the price for agent i is just

$$p_i^{k+1}(\tau) = p_i^k(\tau) + \gamma_i^k[d_i(\tau) + \sum_{i\neq j} C_{ij}\hat{x}_j(k)]$$

■

If we move back to the agents and forget the game theory interpretation, it can be seen that under the market approach agents update their prices and also the prices of their neighbors and therefore there is no need to exchange the state estimate. All

the public information needed are the prices and their updates. The estimation of the agents through the different iterations bring increments or decrements in the prices until equilibrium prices are reached. However, there is a price to pay in terms on the amount of model information that agents have. With this price mechanism it is needed that agent i has knowledge of the terms C_{ji}. In other words, agents have knowledge of the collateral effects they induce in their neighbors.

Remark 16 *From an economic point of view, the situation can be interpreted as a market of help services. The price $p_i(\tau)$ is the unit price that agent i has to pay to his neighbors for them to change their current contribution to his output. The fact that neighbors of an agent i change their estimates affects to his price in such a way that it reflects how costy is for his neighborhood to help him after the estimate update. On the other hand helping his neighbors is rewarded in (4.15). Taking all of this into account, agents are both service offerers and demanders. All of them behave selfishly according to the prices fixed by the* market*, that is, the distributed price mechanism proposed in this approach.*

Remark 17 *In welfare economics, under certain assumptions such as the absence of externalities in transactions, it is proved that market prices guarantee that, despite of agents selfish behavior, a Pareto optimum is achieved [97]. In the optimization problem that we have, unfortunately we have to deal with the presence of externalities, taking this term in a wide sense. That is, decisions taken by agent i also affect other agents. In order to overcome this problem and still reach a Pareto optimum while keeping selfish, i.e. decentralized, behavior in the agents, some modifications have to be introduced in the market: first, all the agents behave as price takers as they were in a competitive market when they really have power to modify the prices and, second, prices are updated globally according to the proposed mechanism.*

4.3 Examples

The problem of estimating the position of a moving object can be faced using different approaches. For outdoor applications in which the precision requirements are low GPS estimation is the most used choice. Radar measurements help to improve the quality of the estimation. When it comes to indoor applications the problem of localization is normally solved by means of a sensor network. In cases in which low precision is needed some it may be enough with a network of presence detectors. If more precision is required then more sophisticated techniques have to be used. In the case that the application is executed in a very controlled scenario, it is possible to use cameras to estimate the position. Infrared or ultrasonic sensors also provide a greater accuracy than the presence detectors. In the last

years the use of the link quality between wireless transceivers has been used too for this kind of applications [49].

4.3.1 Application to mobile robot localization

This subsection is based on the simulation scenario proposed in [31].

Let us consider a system consisting a set of $\mu = \{1, .., M\}$ reference nodes or beacons and a set $\eta = \{1, .., J\}$ of mobile devices. In this example we will consider $M = 6$ beacons and $J = 8$ mobile devices, which are located in the positions depicted in figure 4.1.

The goal is to estimate the position of the moving devices. If the sample time is assumed to be low enough, it is possible to simplify the dynamics considering that the devices move at every sample time a bit with respect their position. The equations for each device are:

$$x_i(\tau + 1) = x_i(\tau) + \Delta x_i(\tau) \; \forall i \in \eta = \{1, .., J\}$$

with $x_i(0) = x_i^0$. The beacon position is fixed so that $x_i(K + 1) = x_i(0) \; \forall i \in \mu = \{1, .., M\}$. The distance between the nodes and the mobile devices can be calculated using

$$d_{ij}^2 = (x_i - x_j)^T (x_i - x_j) \; \forall i, j \in \eta, \mu.$$

The distance can be linearized around the steady state positions $\overline{x}_i$ using a first order Taylor approximation, which leads to

$$d_{ij}^2 = \overline{d}_{ij}^2 + 2(\overline{x}_i - \overline{x}_j)^T (x_i - \overline{x}_j) + 2(\overline{x}_i - \overline{x}_j)^T (\overline{x}_i - x_j)$$

with $\overline{d}_{ij}^2 = d_{ij}^2(\overline{x}_i, \overline{x}_j)$. Now, system variables can be introduced for all the mobile devices such that:

$$\begin{aligned} x_i(\tau) &= x_i(\tau) - \overline{x}_i \; \forall i \in \eta \\ y_{ji}(\tau) &= d_{ij}^2 - \overline{d}_{ij}^2 \; \forall i \in \eta, \forall j \in \eta, \mu \\ C_{ji} &= 2(\overline{x}_i - \overline{x}_j) \; \forall i \in \eta, \forall j \in \eta, \mu. \end{aligned}$$

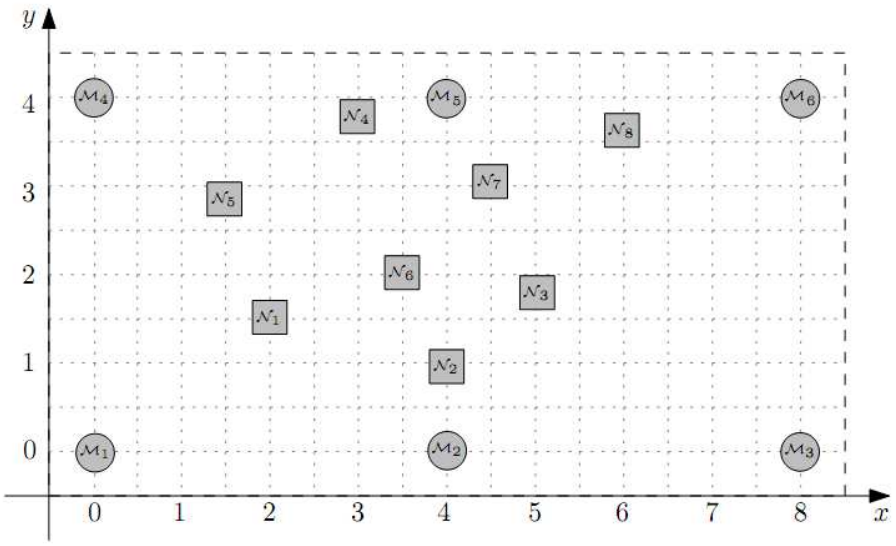


Figure 4.1: Initial situation of the devices.

Each moving device's output provides information about the distance with respect the other moving devices and the beacons. If white gaussian additive noise is assumed in the state and output then each device can be modeled according to equation (4.1).

In order to make the situation more realistic it is assumed that only devices and beacons within a range can communicate. Thus a communication radius ρ is defined. In general two devices i and j can communicate if $d_{ij} < \rho$. A communication graph can be defined to reflect which devices can communicate at each sample time. The communication graph at initial time is given by the following matrices:

$$A_0^\eta = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}, \; A_0^\mu = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

where $A_0^\eta(i,j) = 1$ if the mobile device i is able to communicate with the mobile device j and $A_0^\mu(i,j) = 1$ if the mobile robot i is able to communicate with the beacon j.

The simulations have been done considering a dynamic graph, that is, a situation where the movement of the devices is big enough to guarantee that the communication graph changes with the relinearization of the system. At each time in which the system is relinearized it is necessary not only to update the equations but the information about the last samples that is kept in the agents. Let us assume that in time τ there is a change of linearization point of the system. Then,

$$x_i(t) = \bar{x}_i(t) + x_i(t) - \bar{x}_i(\tau) \; \forall t \in [\tau - N, \tau]$$
$$y_i(t) = C_i(\tau)\hat{x}_i(t) \; \forall t \in [\tau - N, \tau]$$

This change of coordinates in the state estimates and the outputs allow to compute the distributed problem without suffering estimation disturbances after the change of linearization point.

The system has been simulated for 40 time samples with a state noise stronger than the original one. The first 10 samples a centralized Kalman filter is working and the second 20 the distributed strategy. In $t = 20$ and $t = 30$ the system is relinearized. The window size used for the estimation was 4. In blue it is depicted the real trajectory and in red the estimation. The results for the estimation of the position of the mobile devices can be seen in figure 4.2. The overall picture is shown in figure 4.3. The quality of the estimation depends on several parameters. For example, the more iterations are made the better the estimation gets. In this figure it can be seen that the estimation is very precise for most agents.

4.3.2 Application to traffic and speed control

Imagine a scenario where there are reference nodes in roads and streets, and cars are equipped with wireless transceivers. Each car could estimate its position using the link quality indicator from the packets in the reference nodes and other cars as an indicator of the distance. In such set up it is possible to imagine many useful applications that benefit from the algorithm presented in this chapter. Some ideas to take advantage of the position estimation would be to use the information as a way to monitor and control the traffic, or to check if the cars are moving without exceeding the speed limits, or, for example, for cars with free parking places around there to communicate it to the network, reducing search time for neighbors. The price to pay to fully enjoy this possibilities would be a loss of privacy because the network would be aware of the position of all the cars in this hypothetic scenario. An alternative could be to hide the identity of the cars so that applications such as traffic control could be maintained.

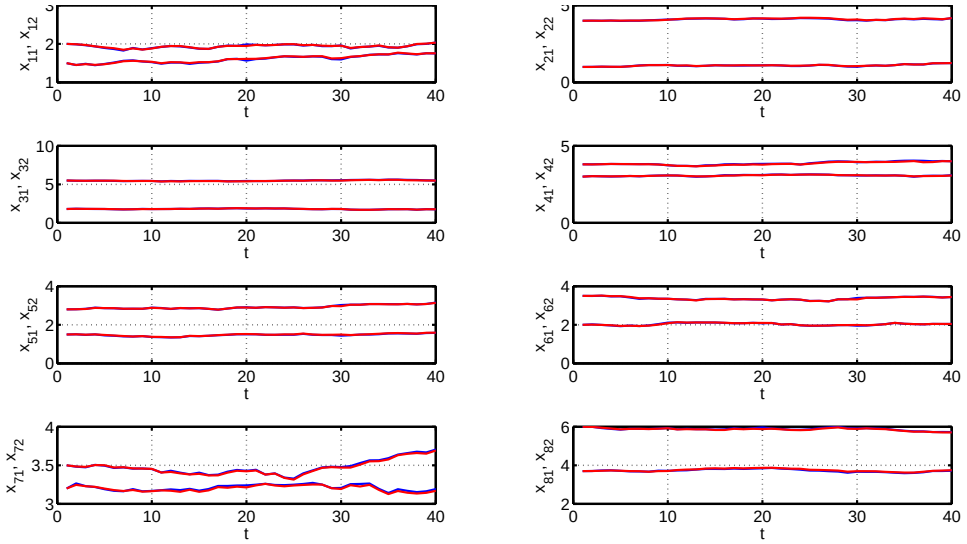


Figure 4.2: Robots' state evolution with noise model mismatch.

In figures 4.5 and 4.4 it can be seen a very simple simulation of what would happen if the mobile robots of the previous subsection scenario would behave as cars going left or right as in a normal road. It is assumed that once a robot has been located it is only necessary to estimate its x component. For this simulation the window size was fixed to 3 and the communication radius to 5. The average number of iterations to reach a 95% of accuracy was 6.27.

4.4 Conclusions

A distributed version of the Kalman filter based on dynamic programming has been developed. The use of dual decomposition allowed the problem distribution. In the simulations presented promising results of the future applications of these techniques are shown.

The different coordination alternatives for the price update that have been presented are also remarkable. In particular, the market approach allows to use dual decomposition without revealing the state of the subsystems. This feature may be interesting in control applications in which dual decomposition is used as well. It will be important for future work some kind of suboptimality bounds to determine the precision obtained in the estimation after a number of iterations. Practical experiments will be developed too to see how the distributed estimation works in real application.

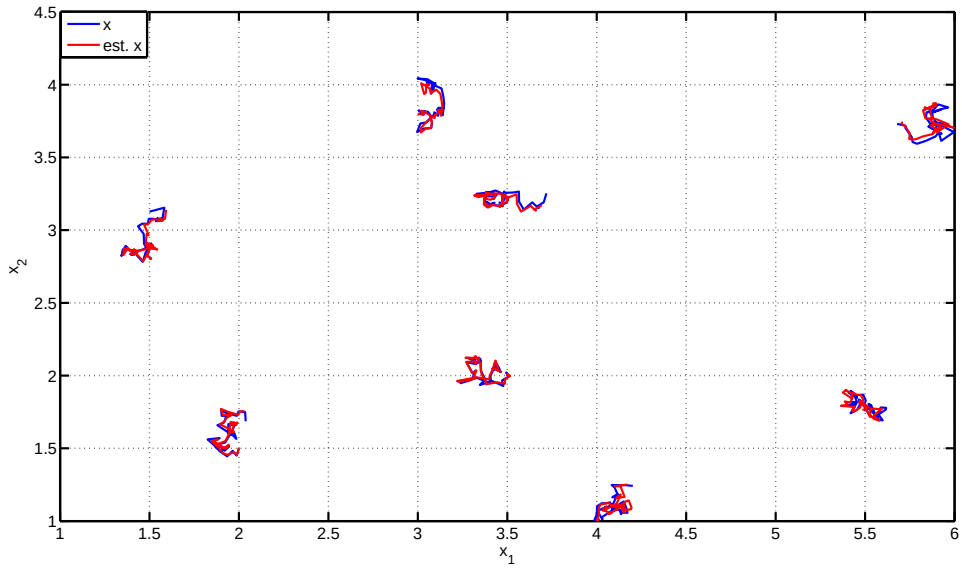


Figure 4.3: Robots' trajectories with noise model mismatch.

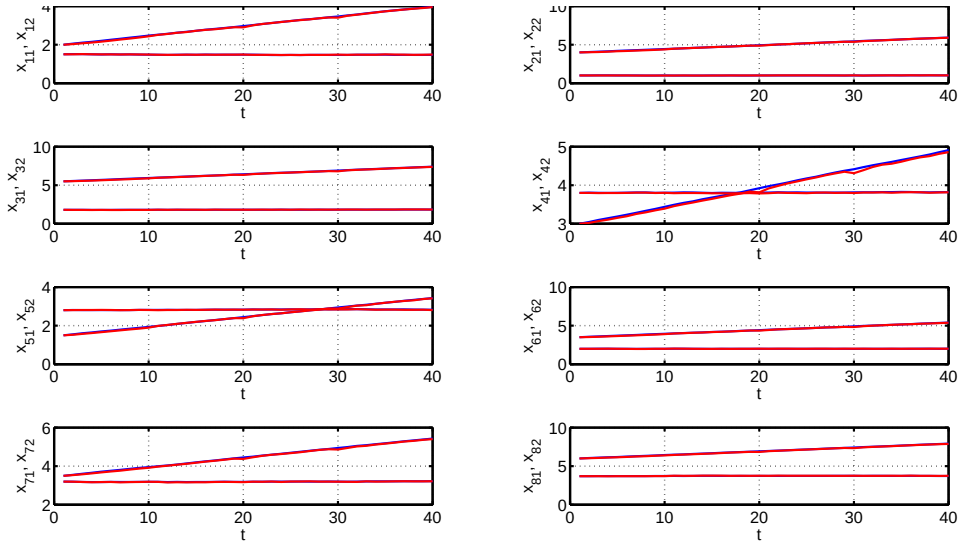


Figure 4.4: Robots' state evolution with noise model mismatch in straight movement.

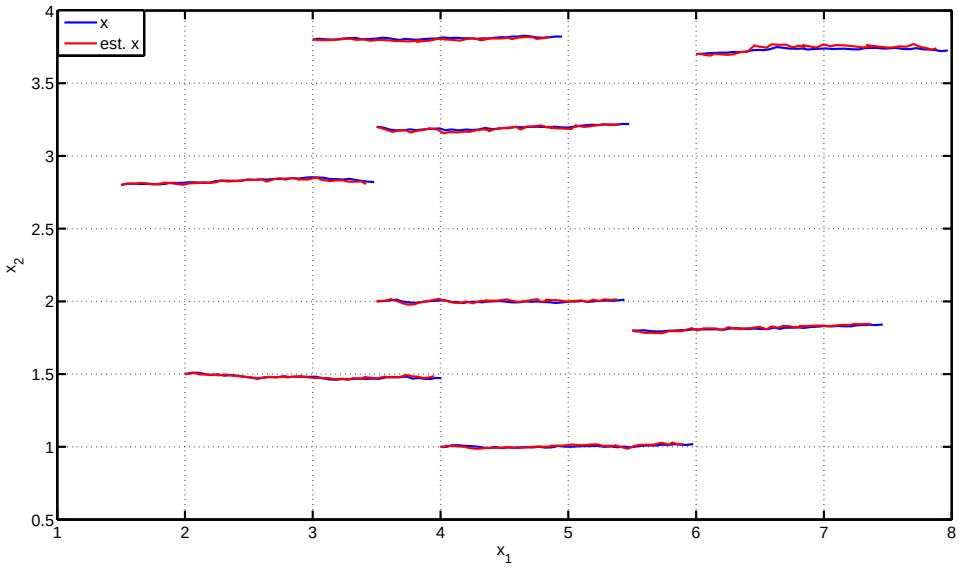


Figure 4.5: Robots' straight trajectories with noise model mismatch.

Chapter 5

Applications of Cooperative Game Theory to the Control and Estimation of Distributed Systems

In the previous chapters we have presented distributed control and estimation schemes in which the division of the system into different subsystems was assumed to be fixed "a priori". In a distributed system, the network configuration imposes constraints on the way agents communicate. However, it is not required that all the agents connected by the network communicate all the time. In fact, broadcast algorithms are avoided if possible. In some cases, it may be better for the agents to separate themselves into different coalitions. In this sense, an interesting topic that is rarely considered in the literature is the evolution of the couplings with time. Decentralized and distributed control schemes often assume that the centralized system is partitioned into a fixed set of low coupled neighborhoods. While the coupling inside a neighborhood is high and demands a coordinated actuation of all its agents, coordination between coalitions is not a major issue. In general, the composition of these neighborhoods is assumed static, that is, the possibility of time varying neighborhoods is not considered. In addition, there are other interesting questions that are not usually addressed such as which elements of a given distributed control system are more critical. Motivated by these issues, in this chapter we focus on distributed systems in which the agents switch between different communication strategies that define which network links are used and we study the underlying properties of a given distributed control scheme using tools from game theory. From a mathematical point of view, game theory is an appropriate framework to study all the phenomena that arise from the mutual interaction of agents that take their decisions alone or in cooperation; see [9, 66].

In particular, we focus on the following three different problems:

- Given a communication network, do all the links have to be enabled all the time? Assuming that there exists a cost for using the communication links, probably at some point it will be preferable to let those agents whose respective subsystems are not highly coupled to work in a decentralized manner. As the coupling effects changes, so does the composition of the neighborhoods.

- Do all the links and the agents have the same relevance in a networked control system? Even when redundancy is one of the major advantages of distributed systems, the consequences of a failure change depending on the link or local controller that fails. It is interesting to determine what agents and links are more relevant in order to take preventive actions if needed. In this way, it would be possible to guarantee a better performance of the overall system in the case of a failure.

- When several agents cooperate to reach a certain objective, do they have to share equally the costs or benefits of the cooperation? This question makes sense specially when the control performance has a direct economical impact. For example, one could think of a power network in which several companies cooperate to provide a service to the final customer. Unless all agents contribute equally, it is not fair to distribute equally the economical benefits of the cooperation.

In this chapter, we study in first place a distributed control scheme in which a set of agents can communicate through a network in order to regulate to the origin a set of unconstrained linear systems by switching between different linear control laws depending on the available information. In second place, it will be seen how to apply these techniques to a state estimation problem. In this context, the application of state estimation schemes to problems in which the state represents the position of an object is very attractive. The localization of moving entities, such as robots or people, is important for many applications. Ideally, there would not be costs or constraints attained to communications. Unfortunately this assumption does not hold in real systems and a trade-off between precision and communicational burden has to be obtained. For this reason, the example chosen to illustrate the concepts that are presented in this chapter is an application in which a set of moving devices try to self localize their own positions.

The chapter is organized as follows. First, some grounds of cooperative game theory are provided. Next, the class of distributed control problems considered is introduced. In the next section example is given to illustrate the results presented in the chapter. Section IV presents the estimation problem and section V presents the estimation simulation example. Finally, in section VI the conclusions of the chapter are shown.

Part of this chapter has been submitted for publication in [56] and it has been presented in [48].

5.1 Cooperative games

In the first chapter of the thesis we introduced some basic concepts of cooperative game theory. In this section we will introduce formally the concepts that will be used in this chapter.

A cooperative game is defined only with two elements, a set $N = \{1, 2, .., n\}$ of different *players* and a function v that assigns a value to each of the 2^N possible coalitions S of agents. In this point we have to remark that $v(S)$ represents the cost to reach the common goal without the assistance of the agents that are not present in the coalition. This definition of cooperative game can be complemented taking into account both the network and the cost associated to the use of the different links. Mathematically a network is defined as a graph (N, L), where L is the set of edges $L \subseteq L^N = \{\{i, j\} | \{i, j\} \subseteq N, i \neq j\}$. Note that this implies that ij and ji represent the same link. The necessary and sufficient condition for any two agents to communicate, and hence cooperate, is that they are at least indirectly connected by the network, that is, there exists a path of active links that connect them. In addition, a cooperative game can also take into account the costs of communication. Therefore, it can be considered that the existence of each link has a fixed cost $c > 0$. With all these ingredients, we can define a cost-extended communication situation H as the tuple (N, v, L, c).

Therefore, the set of players may be partitioned into different coalitions S. It is important to notice that not all the agents in a given coalition S have to be connected by the network L, that is, only those agents in S that are at least indirectly connected will be able to communicate. This fact may cause the partition of the set S into different subsets of agents C that will be called communication components. We will denote by S/L the set of all communication components in a coalition S and by $L(S)$ the links used by that coalition. Note that these concepts can also be applied to the grand coalition N; that is, the coalition composed of all the agents in the game.

As it can be seen, the definition of a game requires to provide a value for each of the 2^N possible coalitions of players. Undoubtedly, this is too much information for any analytical purpose. Thus, it is interesting to have a mathematical tool which provides individual outcomes of the game, that is, a payoff vector that specifies the benefit or cost that each player may reasonably expect from the game. Mathematically, a payoff or allocation vector is defined as $o = (o_i)_{i \in N} \in \mathbb{R}^N$ and specifies for each player i the profit or cost o_i when he cooperates with other players. This is just the role of allocation rules, which are designed to

provide a payoff vector as the expected solution of the cooperative game. However, given a game there are several possible rules to determine a payoff vector as a solution. In this thesis it will be used the Shapley value , which is the only allocation rule $\gamma(N, v)$ that verifies the following properties [66]:

- Efficiency: that is, the payoffs of the players add exactly $v(N)$.

$$v(N) = \sum_{i \in N} \gamma_i(N, v).$$

  In terms of control theory this can be interpreted as a way to distribute the cost of the centralized system between the agents.

- Additivity: let $\gamma(N, v)$ and $\gamma(N, w)$ be two coalitional games, this property implies that

$$\gamma(N, v + w) = \gamma(N, v) + \gamma(N, w).$$

  From the point of view of control theory this is equivalent to have a set of players cooperating in two different goals. The gains from cooperation in one area would be $\gamma(N, v)$ and in the other $\gamma(N, w)$. The result from cooperation in both areas would be described by the game $\gamma(N, v + w)$.

- Symmetry: player that contribute in the same quantity to a given coalition receive the same payoff, that is,

$$\gamma_i(N, v) = \gamma_j(N, v) \leftrightarrow v(S \bigcup i) = v(S \bigcup j) \ \forall S.$$

- Passive-player property: a player that do not contribute marginally to the value of any coalition must not receive anything extra from cooperation, that is:

$$\gamma_i(N, v) = v(i) \leftrightarrow v(S \bigcup i) = v(S) + v(i) \ \forall S.$$

The Shapley value can be interpreted as the payoff vector that gives to each player his expected marginal contribution to a random coalition. The Shapley value for the agent game defined by (N, w^H) is called the Myerson value of the game. In the case of the link game (L, r^H) the Shapley value offers information about the cost of each of the links and it can also be used to construct the so called position value of the game, which is a payoff vector that assigns to each of the agents a value consisting in the sum of half the value the links that are incident to him.

The combination of the Shapley value and cost-extended communication situations will allow us to study several inherent properties of the agents and the network. To this end, we

consider two different games based on the same elements. First, we consider a game (N, w^H) defined by characteristic function $w^H(S)$ which assigns to each coalition the following cost

$$w^H(S) = \sum_{C \in S/L} v(C) + c|L(S)|, \ \forall S \subseteq N, \tag{5.1}$$

where $|L(S)|$ is the number of links that are used in the coalition and c is the link cost. We denote this game as the "agent game". Note that according to (5.1) the value of a coalition S is the sum of the values of its members separated into their corresponding communication components. The analysis of the coalitions in a cost-extended network game provides information about which are the most valuable agents for the system.

The second game considered was proposed by Borm [75] and consists on changing the focus to links instead of agents. The gains or costs from cooperation are attributed to communication links, which lead us to define a cost-extended "link game" as a tuple (L, r^H) associated to the cost-extended communication situation H. The characteristic function for this game is defined as

$$r^H(A) = \sum_{C \in N/A} v(C) + c|L(A)|, \ \forall A \subseteq L, \tag{5.2}$$

which is defined for all the possible subsets A of links contained in the original network L. Note that, in the characteristic function of the link game defined by equation (5.2), the grand coalition is divided into its communication components and therefore its value is the sum of the values of the corresponding components and the cost of the links that are employed for the communication defined by the set A. The analysis of all the coalitions of links provides information about the relevance of each link and show which network configuration is better at a given time instant.

These two games are useful to evaluate two different and important aspects of a distributed system: which agents and links are more relevant at a given time and state. Note that despite the solutions in cooperative games are focused in the obtention of payoff vectors to estimate the distribution of costs or benefits between the players, in this thesis it will be shown that it is possible to use these values as tools for the analysis of relevance of the agents and the links in a distributed control problem.

In the following sections we will show how a distributed system can be characterized by a cost-extended communication situation H. The relation is not straight forward, though. The main difficulty comes from how to define the characteristic function. This function is defined for each possible coalition S in the coalitional game and its calculation requires to determine

the cost that would suppose for the agents in S achieve the goal without the cooperation of the agents out of S for a given communication architecture L.

5.2 Distributed control problem formulation

In this chapter an application of coalitional games to distributed control is proposed. The main objective of the proposed approach is to provide an a priori analysis of the best possible use of the communication network at each sampling time assuming that the use of each link has a cost. In addition, a qualitative interpretation of the results is given so that it can be determined which agents need most to communicate and which links are more important for a given distributed scheme.

We consider a linear system divided in $i = 1, \ldots, N$ subsystems defined by the following model

$$\begin{array}{l} x_i(t+1) = A_{ii}x_i(t) + B_{ii}u_i(t) + d_i(t), \\ d_i(t) = \sum_{j \neq i} A_{ij}x_j(t) + \sum_{j \neq i} B_{ij}u_j(t), \end{array} \tag{5.3}$$

where $x_i \in \mathbb{R}^{q_i}$ and $u_i \in \mathbb{R}^{r_i}$ with $i = 1, \ldots, N$[1] are the states and inputs of each subsystem respectively. The variable $d_i(t)$ is the influence of the neighbors' states and inputs in the update of x_i. Each agent i has access only to its state x_i and decides at each sample time the value of its corresponding input u_i.

We assume that there exists a network L which allows the agents to exchange information. The type of information exchanged depends on the distributed control algorithm that is being used. Any two agents that are not indirectly connected by the network will not be able to exchange any type of information.

The control objective is to regulate the state of all the subsystems to the origin while minimizing a cost which depends on the state and input trajectories and on the communications. This cost will be used to define the characteristic function of a cooperative game. The stage cost of each agent is defined as follows

$$\ell_i(t) = x_i^T(t)Q_ix_i(t) + u_i^T(t)R_iu_i(t).$$

The objective is to minimize the total cumulated cost taking into account the communication costs defined in the previous section. In some applications the stage cost can be interpreted in economic terms.

[1] In this chapter, the letter N stands for the number of players in the game, not the horizon used in the MPC schemes.

The communication costs depend on the number of links that are being used. Note that given an initial network L not all the links have to be used all the time. The term *network mode* will be used to denote each of the possible subset of links A in L. The change of performance for the control system will be analyzed for the case in which some of the links were not *present*. The analysis of the cost trade-off between control performance and communicational costs will determine which is the best network mode A at state x.

Remark: The value of the cost of the use of a communication link during a sample time has to be determined *ad hoc*. For example, in a wireless network this value can be function of the inverse of the remaining battery of the nodes that support the link. In case that the characteristic function of the games can be interpreted in economic terms then it could be calculated the actual cost of transmissions through the link. In general, a simple way to provide a value is to assign is to average the cost impact between enabling or disabling the link in the system or just to set a bound on cost improvement for the link to be enabled.

Remark: In systems where the number of agents is too big the calculations can be simplified assuming that $w(U \bigcup V) = w(U) + w(V)$, that is, the value of the coalition of the players in the set $U \bigcup V$ is equal to the sum values of U and V. This approximation is much better when U and V are in different communication components.

Remark: Note that initially the characteristic function for a game defined from a distributed control problem should be subadditive, that is $w(S) + w(T) \leq w(S \bigcup T)$, that is, the control performance gets better as there are more agents involved. However there is a hidden implication in the last property: communication is costless. Theoretically the last statement can be a good starting point to develop some results, but in practice this results to be a fallacy.

5.2.1 Distributed control algorithm

In this section we present a distributed control scheme that at each sampling time, implements a certain communication strategy defined by a network mode A. The communication strategy A is chosen every D sample times. To this end, the agents must broadcast their state and take a decision about the communication strategy that will be used in the next D time steps. This leads to a double sample rate control system. As a result of this policy, the agents are separated into separated groups C that are able to communicate defined as *communication components*. We will denote by N/A the set of all communication components in which the set N is partitioned. Note that $\bigcup_{\forall i \in N/A} C_i = N$ and $C_i \bigcap C_j = 0$ for all $i \neq j$.

We assume that for each network mode A, a different controller that stabilizes the whole

system which takes into account which agents can communicate is defined. The details about the calculation of the controller will be presented later in this section. In particular, we assume that each communication component $C \subseteq N/A$ implements a linear controller

$$u_C = K_C^A x_C,$$

where $u_C \in R^{\sum_{i \in C} r_i}$ is the input of a given communication component defined as $u_C = \{u_i\}_{i \in C}$, $x_C \in R^{\sum_{i \in C} q_i}$ is the the state of a given communication component defined as $x_C = \{x_i\}_{i \in C}$ and K_C^A is the matrix which defines the controller implemented by the communication component for the network mode A.

From the set of the matrices K_C^A, the following centralized linear controller for the overall system, characterized by the absence of communication for agents in different communication components can be obtained

$$u = K_A x$$

where $u \in R^{\sum_{i \in N} r_i}$ is the input of the centralized system defined as $u = \{u_i\}_{i \in N}$, $x \in R^{\sum_{i \in N} q_i}$ is the the state of the centralized system defined as $x = \{x_i\}_{i \in N}$ and K_A is the matrix which defines the centralized controller implemented for the network mode A.

Note that matrix K_A takes into account the communications constraints in A. For example, if the i-th element of u and the j-th element of x belong to different communication components, then $K_A(i,j) = 0$; that is, the i-th input does not depend on the j-th state. For the particular case in which each communication component is composed by systems with consecutive numeration in the set N, then $K_A = diag(K_{C_1}^A, K_{C_2}^A, \ldots)$. Note that all the matrices K_C^A have to be designed so that K_A guarantees closed-loop stability for the centralized system. If a given mode is not able to stabilize the system, then it is not taken into account.

In order to decide which communication strategy must be implemented, we assume that there exists a quadratic function that satisfies

$$x^T P_A x \geq \sum_{j \in N} \sum_{k=0,..,\infty} \ell_j(k) \tag{5.4}$$

that is, P_A is a weight matrix that provides an upper bound of the cost to infinity of the centralized system in closed-loop with the controller $u(k) = K_A x(k)$ starting from the initial state $x(0) = x$. This quadratic function will be used so decide the optimal communication mode as well as to define the link problem.

Summing up, the proposed distributed control scheme is implemented as follows:

1. If the sample time is a multiple of D, all the agents broadcast their state and calculate which is the network mode A that minimizes the function $r^H(A,x)$. Otherwise, each agent sends his state only to those agents that belong to his communication component.

2. Each agent uses the state information received in order to update its control action using its corresponding communication component feedback matrix K_A^C. Globally, this implies that the linear controller $u = K_A x$ is applied.

Remark: It is possible to establish more sophisticated assumptions about the system. However, it has been preferred to simplify the system as much as possible in order to focus on the three mentioned problems. The definitions done are enough to identify both the agent and link games and to solve the decision problem of what agents should communicate at each sample time.

Remark: The ideas presented in this chapter can also be applied to different distributed control strategies (such as distributed MPC schemes) or to more complex systems (such as non-linear systems). In that case, an appropriate definition of the utility function has to be provided. We propose to use a bound of the cost-to-go of the different modes, but other approaches are also possible.

5.2.2 Network modes

In order to analyze which agents should communicate, the link game associated to the cost-extended communication situation H of the current state x is studied. The characteristic function that assigns a value to each communication mode A is based on the upper bound of the cost-to-go of all the communication components in the network and the corresponding communication costs and is defined as

$$r^H(A, x) = x^T P_A x + c|L(A)|, \ \forall A \subseteq L.$$

The best possible communication mode provides the winner network configuration choice. This mode is obtained for a given state x by minimizing $r^H(A, x)$ over A. The function $r^H(A, x)$ will be also used to define the link game.

We introduce next the concept of dominance between modes. The mode A dominates the mode B if $r^H(A, x) < r^H(B, x)$. In general, we say that mode i is dominant if the last inequality holds $\forall B \neq A \subseteq L$. Therefore, the set of points CR_A for which a network mode A is dominant is characterized by the states such that A provides the best (lower) cost, that is,

$$CR_A = \{x \in R | r^H(A, x) \leq r^H(B, x), \forall B \subseteq L\}.$$

The union of all these regions covers the whole state space. The distributed controller will switch between the different modes as the state moves from one region to another. In order to check online which is the optimal network mode for the particular distributed control

scheme presented, it is sufficient to evaluate the quadratic functions which define the cost of each mode. Note that in general, the regions associated to the network modes are non convex. For more complex cases it may not be possible to calculate a explicit characteristic function $r^H(A, x)$ for each network mode and the application of techniques might be necessary to find off-line the regions of each of the modes in order to check the optimal mode. Suboptimal approaches based on exhaustive simulation of all the possible network modes can be used to find these regions of dominance. For the distributed control scheme considered, the boundaries of the regions are defined by quadratics which depend on the different weight matrices P_A and number of links $|L(A)|$ which define each mode. In particular, it is possible to calculate explicit frontiers. To do so, let $J_i = x^T P_i x + c_i$ and $J_j = x^T P_j x + c_j$ be the costs for network modes i and j respectively. Mode i is chosen if $J_i < J_j$ and viceversa. Thus, the frontier between the regions of dominance of networks modes i and j is given by the following equation:

$$\begin{aligned} &J_i = J_j \\ &x^T P_i x + c_i = x^T P_j x + c_j \\ &x^T (P_i - P_j) x + c_i - c_j = 0 \end{aligned} \tag{5.5}$$

The shape of the bound will depend on the matrix $P_i - P_j$, which in general does not have to be positive definite.

5.2.3 Link analysis

The link game is constructed by a set of players composed by the links of the network L and a characteristic function that assigns to each communication mode A an given utility. In this case, we propose to use $r^H(A, x)$ to define the link game. Once the link game is constructed, a qualitative and quantitative analysis of the relevance of the links in the game may be obtained from the corresponding Shapley value $\gamma(L, r^H, x)$. Each component of this vector represents the cost of a given link for the system. In other words, the lower value a link has, the higher utility it has for the system.

5.2.4 Agent analysis

A qualitative and quantitative analysis of the relevance of the agents of the system may be obtained from the Shapley value $\gamma(N, w^H, x)$ of the corresponding agent game. To build such game it is necessary to define the characteristic function that assigns a value to each coalition S of agents for a given network L. To this end, it is not possible to use the controllers defined for each communication component for a given network mode A because those controllers take into account the particular communication constraints of A. For this reason, for each communication component of L, we define a controller K_C and a weight

matrix P_C such that

$$x_C^T P_C x_C \geq \sum_{j \in C} \sum_{k=0,..,\infty} \ell_j(k) \tag{5.6}$$

where x_c is obtained from the current state x and is composed of the states of all the subsystems that belong to C; P_C is a weight matrix that provides an upper bound for the cost to infinity of the systems that belong to C in closed-loop with the controller $u_C(k) = K_C x_C(k)$ starting from the initial state $x_C(0) = x_C$ assuming that all the inputs and states that belong to agents outside the coalition are zero.

Then, the characteristic function of the agent game that defines the utility of a coalition S is defined as

$$w^H(S, x) = \sum_{C \in S/L} x_C^T P_C x_C + c|L(S)|, \ \forall S \subseteq N.$$

The Shapley value of the game (N, v^H, x) provides concise information about the relevance of all the agents in the game. The lower the value the is, the more relevant role the agent has in the game. It is important to stand out that the Shapley value of the agent game, as it is defined, does not have any physical meaning.

Remark: This assumption allows only to calculate a simple approximation of the Shapley value of the agent game. More conservative choices could have been made, for example the agents outside of the coalition could have been considered as disturbances and then a min-max approach used. Nevertheless, we must not forget that the goal of all the agents is to regulate the system to the origin. Moreover, with the design method presented in the next subsection it is possible to calculate feedback gains such that the coalitions C and $N - C$ are able to stabilize the overall system. For this reason, we consider the approximation made appropriate.

5.2.5 Coalitional game design method

In this section we present a method to design for a given system (5.3) all the matrices that define the controllers and the weights of the upper bound functions for each of the networks modes (K_A and P_A for all $A \subseteq L$) as well as each of the possible communication components of the agent problem (K_C and P_C for all $C \subseteq N/L$). To this end, we present two different theorems.

First, we deal with the problem of finding the matrices that define the controllers and the weights of the upper bound functions for both each of the possible communication components of the agent problem (K_C and P_C for all $C \subseteq N/L$). In this case, K_C must stabilize the states of C assuming that the outputs and inputs that do not belong to that communication

component are zero. In addition, P_C must guarantee that (5.16) holds. The following theorem presents an LMI constraint that can be used to solve this design problem.

Theorem 7 *Let $C \in S/L$ be a set of independent communication components for a given communication situation H whose dynamics are given by $A_C = \{A_{ij}\}$, $\forall i,j \in C$ and $B_C = \{B_{ij}\}$, $\forall i,j \in C$ and its stage cost defined by $Q_C = diag(Q_i)$ and $R_C = diag(R_i)$, $\forall i \in C$. If there exist matrices W_C and Y_C such that the following constraint is satisfied*

$$\begin{bmatrix} W_C & W_C A_C^T + Y_C^T B_C^T & W_C Q_C^{1/2} & Y_C^T R_C^{1/2} \\ A_C W + B_C Y & W_C & 0 & 0 \\ Q_C^{1/2} W_C & 0 & I & 0 \\ R_C^{1/2} Y_C & 0 & 0 & I \end{bmatrix} > 0 \tag{5.7}$$

then matrices $P_C = W_C^{-1}$ and $K_C = Y_C W_C^{-1}$ satisfy (5.16) and stabilize the states of C assuming that the outputs that do not belong to that communication component are zero.

Proof: Applying iteratively backwards Schur's complement to equation (5.7) and taking into account the proposed variable change, it can be seen that if (5.7) holds then the following constraint can be obtained

$$(A_C + B_C K_C)^T P_C (A_C + B_C K_C) - P_C + Q_C + K_C^T R_C K_C \leq 0 \tag{5.8}$$

This constraint guarantees that the system defined by matrices A_C, B_C is stable in closed-loop with a the linear controller defined by K_C. In addition, pre and post multiplying (5.8) by $x_C(k)$ we obtain the following inequality

$$x_C(k+1)^T P_C x_C(k+1) - x_C(k) P_C x_C(k) + \sum_{j \in C} L_j(k) \leq 0$$

Summing the previous inequality from $k = 0$ to $k = \infty$ and assuming that $\lim_{k\to\infty} x_C(k) = 0$ (recall that the closed-loop system is stable) we obtain that

$$x_C(0) P_C x_C(0) \geq \sum_{k=0}^{\infty} \sum_{j \in C} L_j(k).$$

■

In the agent game the approximation of the cost of the communication component C is based on the cost to infinity given by the $x_C^T P_C x_C$. This upper bound is calculated assuming that the rest of the agents states and inputs are zero. The agent game only provides grounds for distributing the benefits or costs between the agents during the game and the resulting feedback gains are never used to control the system, so it is not necessary to impose centralized stability as a requirement.

The link game is based on a different point of view of the communication situation. In this case the grand coalition controls the system taking into account all the possible network configurations $A \subseteq L$. Each network configuration A divides the system in a set of communication components $C \in N/A$. In this case the stability of the centralized system has to be guaranteed because these linear feedback will be applied to control the system.. Following the same approach as in the agent game, the following theorem is presented.

Theorem 8 *Let $A \in L$ be a set of active links for a given communication situation H. The dynamics of the whole system are given by $A_N = \{A_{ij}\}$, $\forall i,j \in N$ and $B_N = \{B_{ij}\}$, $\forall i,j \in N$ and its stage cost defined by $Q_N = diag(Q_i)$ and $R_N = diag(R_i)$, $\forall i \in N$. If there exist matrices $W_N = \{W_{ij}\}$, $\forall i,j \in N$, where $W_{i,j} \in \mathbb{R}^{q_i \times q_j}$, and $Y_N = \{Y_{ij}\}$, $\forall i,j \in N$, where $Y_{i,j} \in \mathbb{R}^{r_i \times q_j}$, such that the following constraints are satisfied*

$$\begin{bmatrix} W_N & W_N A_N^T + Y_N^T B_N^T & W_N Q_N^{1/2} & Y_N^T R_N^{1/2} \\ A_N W_N + B_N Y_N & W_N & 0 & 0 \\ Q_N^{1/2} W_N & 0 & I & 0 \\ R_N^{1/2} Y_N & 0 & 0 & I \end{bmatrix} > 0 \tag{5.9a}$$

$$\begin{aligned} &s.t. \\ &W_{ij} = 0, Y_{ij} = 0 \; \forall i,j \text{ such that } x_i \in C, x_j \notin C \end{aligned} \tag{5.9b}$$

then matrices $P_A = W_N^{-1}$ and $K_A = Y_N W_N^{-1}$ satisfy (5.4), all the communication constraints imposed by the network mode A and stabilize the whole system.

Proof: The proof follows the same reasoning as the proof of Theorem 9. In this case the LMI constraint (5.9b) and the proposed variable change guarantee that the following inequality holds

$$(A_N + B_N K_A)^T P (A_N + B_N K_A) - P_A + Q_N + K_A^T R_N K_A \leq 0. \tag{5.10}$$

Stability and (5.4) follow.

The constraints imposed to the LMI guarantee that K_A and P_A satisfy the communication restrictions of network mode A. Let P_A and K_A be descomposed in blocks analogously to

W_N and Y_N, that is $P_A = \{P_{ij}^A\}$, where $P_{ij}^A \in \mathbb{R}^{q_i \times q_j}$, and $K_A = \{K_{ij}^A\}$, where $K_{i,j}^A \in \mathbb{R}^{r_i \times q_j}$. Without loss of generality let us assume that all the states of the subsystems that belong to a same communication component have been grouped, that is, P_A can be written as a block diagonal matrix $P_A = diag(P_C)\ \forall C \in N/A$, where $P_C = \{P_{ij}^A\}$ such that $i, j \in C \in N/A$. Given that the inverse of a block diagonal matrix another block diagonal matrix in which the original blocks are inverted, that is $W_N = P_A^{-1} = diag(P_C^{-1})$, it can be concluded that $W_{ij} = 0\ \forall i, j$ such that $x_i \in C \in N/A, x_j \notin C \in N/A$ implies that $P_{ij}^A = 0\ \forall i, j$ such that $x_i \in C \in N/A, x_j \notin C \in N/A$. Reordering the states in communication components also allows to write K_A as a block diagonal matrix $K_A = diag(K_C)\ \forall C \in N/A$, where $K_C = \{K_{ij}^A\}$ such that $i, j \in C \in N/A$. As K_C and P_C are dimensioned for the same state x_C then $Y_N = K_A W_N = diag(K_C P_C^{-1})\ \forall C \in N/A$. Thus, the fact that $Y_{ij} = 0 \forall i, j$ such that $u_i \in C \in N/A, x_j \notin C \in N/A$ is equivalent to make $K_{ij} = 0 \forall i, j$ such that $u_i \in C \in N/A, x_j \notin C \in N/A$. ■

Remark: This theorem can be used also to provide a cost approximation for the agent game following a coherent criterium for all the agents that are not in communication component C. For example, the matrices P_C could be calculated assuming that all agents outside C work in a decentralized manner.

5.3 Distributed control simulation results

In this section we show an academic example that illustrates the concepts and techniques presented in the chapter. The distributed system considered in the example is shown in figure 5.1. It consists of four agents, represented by boxes, which are coupled by pairs (the coupling interactions are represented by arrows). For example, agent 1 disturbs agents 2 and 3 and is

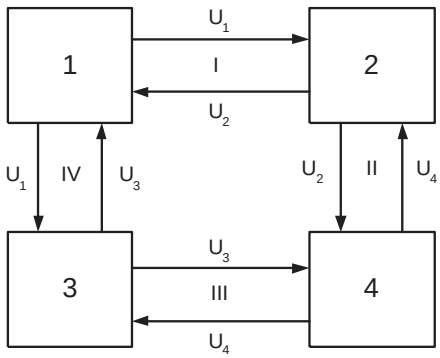


Figure 5.1: Four systems coupled through the inputs.

also disturbed by these two agents. The matrices that define the system are the following:

$$
\begin{aligned}
A_{11} &= \begin{bmatrix} 1 & 0.8 \\ 0 & 0.7 \end{bmatrix} B_{11} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} B_{12} = \begin{bmatrix} 0 \\ 0.15 \end{bmatrix} B_{13} = \begin{bmatrix} 0 \\ 0.15 \end{bmatrix} \\
A_{22} &= \begin{bmatrix} 1 & 0.6 \\ 0 & 0.7 \end{bmatrix} B_{21} = \begin{bmatrix} 0 \\ 0.15 \end{bmatrix} B_{22} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} B_{24} = \begin{bmatrix} 0 \\ 0.15 \end{bmatrix} \\
A_{33} &= \begin{bmatrix} 1 & 0.9 \\ 0 & 0.8 \end{bmatrix} B_{31} = \begin{bmatrix} 0 \\ 0.15 \end{bmatrix} B_{33} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} B_{34} = \begin{bmatrix} 0 \\ 0.15 \end{bmatrix} \\
A_{44} &= \begin{bmatrix} 1 & 0.8 \\ 0 & 0.5 \end{bmatrix} B_{42} = \begin{bmatrix} 0 \\ 0.15 \end{bmatrix} B_{43} = \begin{bmatrix} 0 \\ 0.15 \end{bmatrix} B_{44} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\
A_{ij} &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \ \forall i \neq j
\end{aligned}
\tag{5.11}
$$

where $x_i \in \mathbb{R}^2$ with $i \in \{1, .., 4\}$ are the states of each subsystem and $u_i \in \mathbb{R}$ with $i \in \{1, .., 4\}$ are the corresponding inputs. The stage costs ℓ_i of all the subsystems are defined by matrices $Q_i = diag(1, 1),\ R_i = 1$ with $i = \{1, 2, 3, 4\}$.

In order to implement the proposed control scheme, matrices K_A and P_A have to be designed for each of the possible modes. Figure 5.2 shows the set of network modes for which each link is enabled. The number of possible modes is 16 and they have been numbered from 0 to 15. For example, in mode 0 no link is enabled while in mode 5 links I and II are enabled.

Modes 11 to 15 have been omitted in the figure and only appear in the legend. These modes constitute all the cases where the grand coalition is formed, that is, there are at least 3

links enabled which allows the agents to have full state information. All these five cases have been grouped in mode number 11 and are considered as a single mode for the purposes of this example. Although there are some differences that deserve to be remarked. First, mode number fifteen has an unnecessary link since only 3 links are needed for full communication between the agents. For this reason the control system would never put this mode into play, which is a logical consequence of the fact that only indirect connectivity between nodes is required. Another important issue is that although modes from 11 to 14 may have the same cost associated, they are not equally preferable. Using the techniques presented in this chapter, it is possible to provide an order of preference between all these modes because not all individual links are equally relevant for the system.

For each mode, a different LMI problem designed according to Theorem 10 have been solved to obtain the corresponding matrices K_A and P_A using Matlab's LMI toolbox. For example, for mode 4, which corresponds to the case in which agents 1 and 3 communicate and coordinate their actions. The resulting matrices are:

$$K_4^T = \begin{bmatrix} -0.25 & 0.00 & 0.02 & 0.00 \\ -0.53 & 0.00 & 0.06 & 0.00 \\ 0.00 & -0.26 & 0.00 & 0.00 \\ 0.00 & -0.45 & 0.00 & 0.00 \\ 0.01 & 0.00 & -0.23 & 0.00 \\ 0.05 & 0.00 & -0.63 & 0.00 \\ 0.00 & 0.00 & 0.00 & -0.27 \\ 0.00 & 0.00 & 0.00 & -0.43 \end{bmatrix}$$

$$P_4 = \begin{bmatrix} 4.56 & 5 & 0 & 0 & -0.36 & -1.1 & 0 & 0 \\ 5 & 9.61 & 0 & 0 & -0.8 & -2.48 & 0 & 0 \\ 0 & 0 & 5.48 & 5.14 & 0 & 0 & 0 & 0 \\ 0 & 0 & 5.14 & 8.34 & 0 & 0 & 0 & 0 \\ -0.36 & -0.8 & 0 & 0 & 4.17 & 5.08 & 0 & 0 \\ -1.1 & -2.48 & 0 & 0 & 5.08 & 11.69 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 5.37 & 5.44 \\ 0 & 0 & 0 & 0 & 0 & 0 & 5.44 & 8.40 \end{bmatrix}$$

It can be seen that K_A satisfies the communication constraints of mode 4.

Once the matrices P_A that define the upper-bound on the different cost-to-go values are obtained, it is possible to determine the optimal network mode for a given state. In addition, it is possible to partition the state space in regions associated to different modes. In order to visualize the boundaries of these sets, we restrict our attention to changes in the state x_1

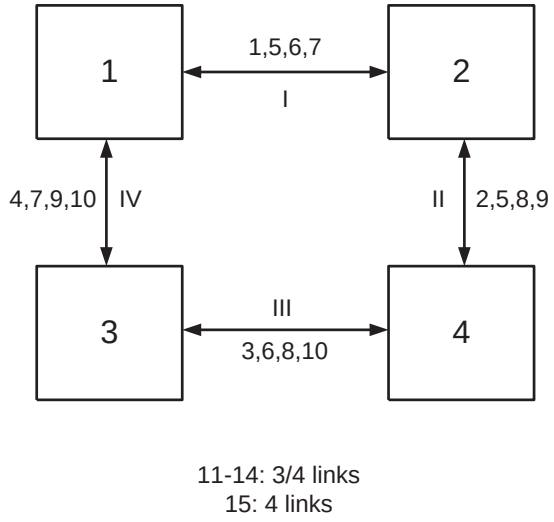


Figure 5.2: Links enabled in each mode.

while the rest of subsystems states are set at the origin. The communication cost is set to $c = 0.5$. In figures 5.3 and 5.4 it can be seen how for values of x_1 far from the origin it is better to apply a centralized mode. The cooperation of all the agents is needed to regulate x_1 to the origin. As state x_1 gets closer to the origin, the recommended mode is number 7, which means that cooperation of agents 1,2 and 3 is recommended. When x_1 gets closer then mode number 4 is applied; only agents 1 and 3 have to cooperate. Finally, as x_1 is around the origin mode 0 is used, that is, all agents can work in a decentralized manner. Finally, in figure 5.5 we restrict our attention to the frontier between network modes 0 and 4 as a function of x_1, which is a ellipse.

Using the matrices P_A, the link game can be constructed for a given state x in order to analyze which links are more relevant. The set of players for this game is defined by the links enumerated by roman letters in figure 5.1. Note that a coalition of links imply a different network configuration mode A, which is equivalent to consider that some of the links in the original network L are disabled. The characteristic function for each of the possible players of the link game for the state

$$x_1 = \begin{bmatrix} 4 \\ 3.6 \end{bmatrix} x_2 = \begin{bmatrix} 2.1 \\ -3 \end{bmatrix} x_3 = \begin{bmatrix} 0.4 \\ 0.8 \end{bmatrix} x_4 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \tag{5.12}$$

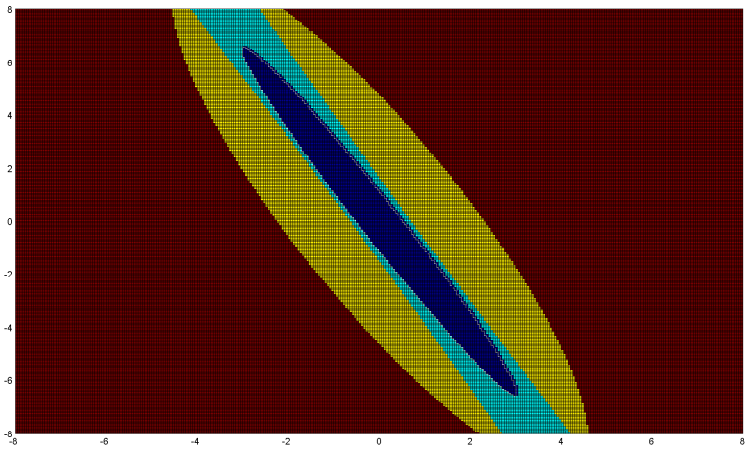


Figure 5.3: Modes as a function of x_1 for $x_2 = x_3 = x_4 = 0$.

is the following

$$
\begin{aligned}
&v(I) = 440.84\\
&v(II) = 437.61\\
&v(III) = 439.24\\
&v(IV) = 364.54\\
&v(I, II) = 402.93\\
&v(I, III) = 439.74\\
&v(I, IV) = 358.35\\
&v(II, III) = 430.53\\
&v(II, IV) = 361.81\\
&v(III, IV) = 365.95\\
&v(I, II, III) = 354.01\\
&v(I, II, IV) = 354.01\\
&v(I, III, IV) = 354.01\\
&v(II, III, IV) = 354.01\\
&v(I, II, III, IV) = 354.51
\end{aligned}
$$

These values show that the optimal network mode A for this state is any of the four composed by three links. It is important to note that 3 links are enough to guarantee communication between all the agents because the only condition for communication between to agents is that they must be at least indirectly connected. For this reason all the coalitions with 3 links enabled have the same value. The Shapley value for this game is

$$\gamma(L, r^H, x) = \begin{bmatrix} 98.94 & 96.91 & 104.28 & 54.36 \end{bmatrix}.$$

This payoff vector guarantees that the players (links) are the responsible of the costs or

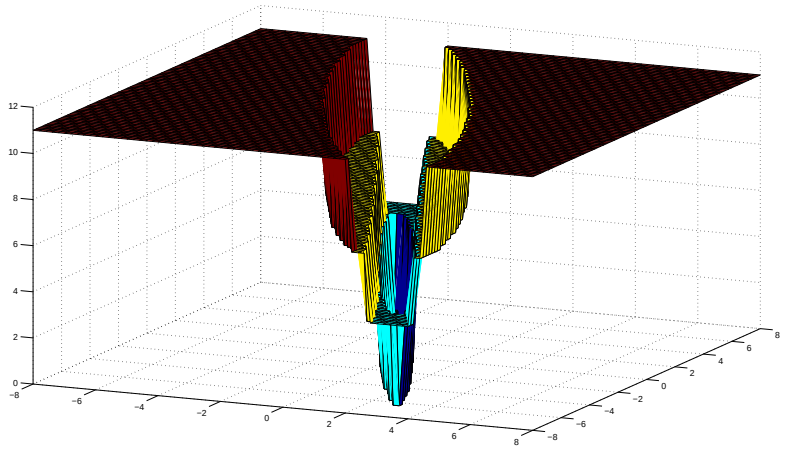


Figure 5.4: Modes (z-axis) as a function of x_1 for $x_2 = x_3 = x_4 = 0$.

benefits the grand coalition gets. This implies that the higher value a link has, the less useful for the system is. It is clear that the link that is more necessary is link number IV, the one that connects agents 1 and 3, and for this reason this link has the lowest value. On the other hand link number III, the one that connects agents 3 and 4, is the one that contributes less to the global objective, and so it is logical that it has the highest value: it gets benefits from cooperation but its contribution is not high in comparison with the other links.

An analogous procedure can be made for the agents, but before matrices K_C and P_C have to be obtained for each of the possible communication components in N/L. In this case we use Theorem 9 to obtain the appropriate matrices for the 15 possible coalitions/communication components. For example, the matrices K_C and P_C for the only communication component in coalition $S = \{1, 2\}$ are:

$$P_C = \begin{bmatrix} 4.57 & 5.02 & -0.48 & -0.86 \\ 5.02 & 9.67 & -1.11 & -1.99 \\ -0.48 & -1.11 & 5.26 & 4.75 \\ -0.86 & -1.99 & 4.75 & 7.63 \end{bmatrix}$$

$$K_C = \begin{bmatrix} -0.25 & -0.53 & 0.02 & 0.05 \\ 0.01 & 0.03 & -0.26 & -0.44 \end{bmatrix}.$$

Using the set of controllers designed, we evaluate the characteristic function for the agent

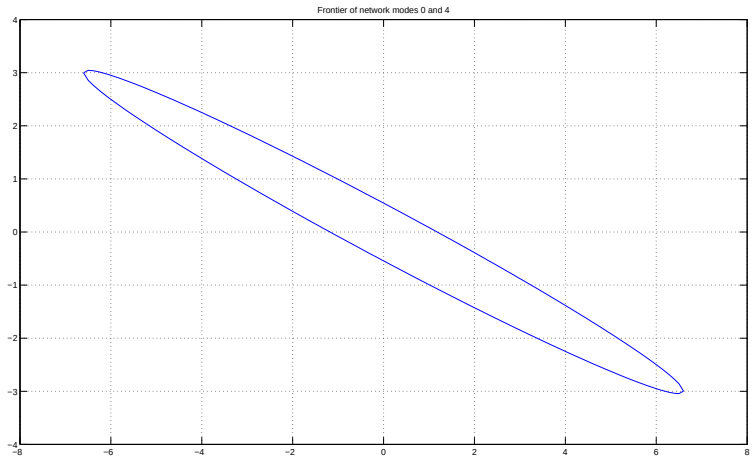


Figure 5.5: Frontier for network modes 0 and 4 as a function of x_1.

game for state x:

$$
\begin{aligned}
v(1) &= 391.12 \\
v(2) &= 35.99 \\
v(3) &= 13.2222 \\
v(4) &= 0 \\
v(1,2) &= 427.61 \\
v(1,3) &= 328.55 \\
v(1,4) &= 391.12 \\
v(2,3) &= 49.21 \\
v(2,4) &= 33.26 \\
v(3,4) &= 12.13 \\
v(1,2,3) &= 358.35 \\
v(1,2,4) &= 389.70 \\
v(1,3,4) &= 329.96 \\
v(2,3,4) &= 39.40 \\
v(1,2,3,4) &= 354.01
\end{aligned}
$$

If we calculate the Shapley value for this game the following vector is obtained

$$\gamma(N, w^H, x) = \begin{bmatrix} 349.89 & 28.46 & -19.07 & -5.26 \end{bmatrix}.$$

The Shapley value is helpful from two points of view. The sum of its components adds up exactly the value that the grand coalition, that is the coalition formed by agents $1, 2, 3$ and 4, has assigned in the game. In cases where the characteristic function has an economic meaning this is very helpful because it provides a possible allocation vector to distribute the profits and benefits from cooperation. Actually, this is sometimes the case when using control

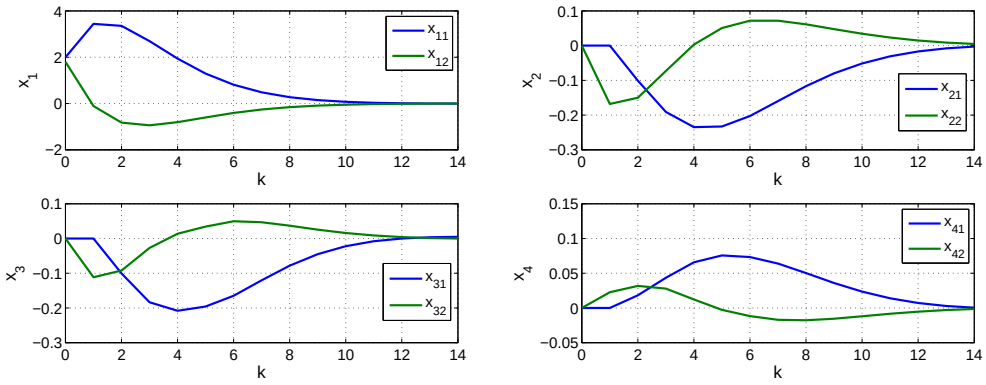


Figure 5.6: States trajectories.

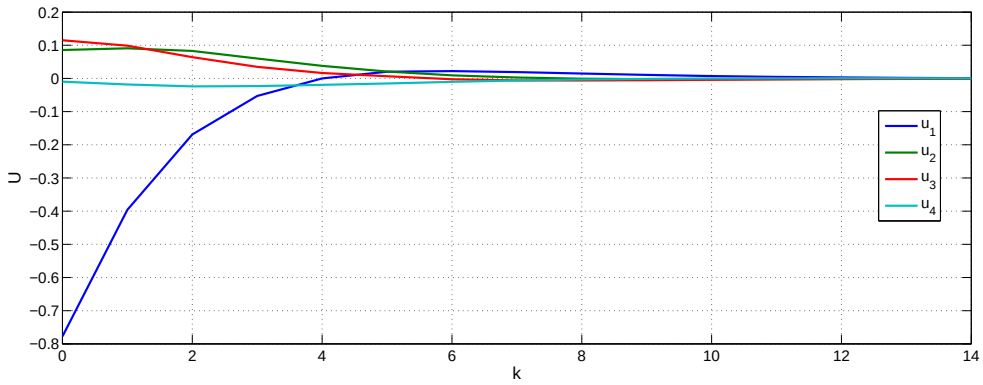


Figure 5.7: Input trajectories.

techniques as model predictive control, where the cost function may represent an economic value. The second utility of the Shapley value is the fact that it shows from a qualitative point of view which agents are more benefited from communication or, in other words, which agents have greater need of communication and help from their neighbors. In this example it is clear that the Shapley value is much greater for agent 1, something logical since is the one furthest from the origin, so he has to assume most of the costs. Agent 2 is also far, but much closer than 1, and so he assumes a lower cost. Agent 3 is also not at the origin but its cooperation is important for agent 1 and this is why he receives a negative cost, that is a profit, because his contribution is greater than his own costs. Finally agent 4 is initially at the origin and so he receives also a reward for his cooperation, less than agent 3 because his help is specially useful to agent 2, which is certainly close to the origin.

The two coalitional games presented in this subsection can be calculated easily once

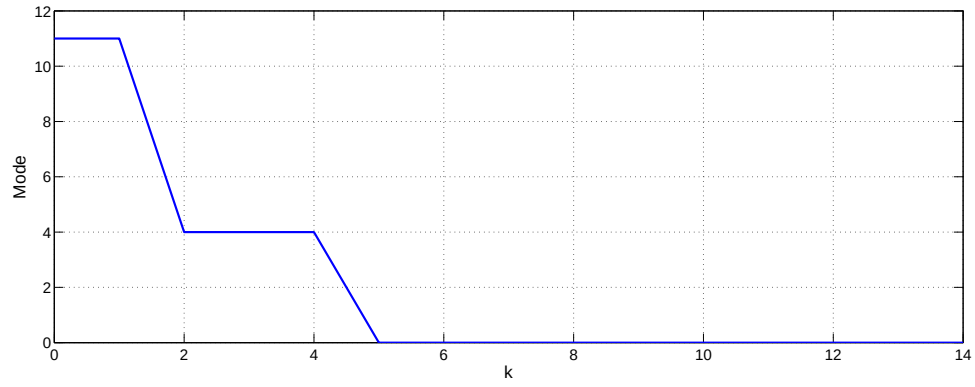


Figure 5.8: Network modes.

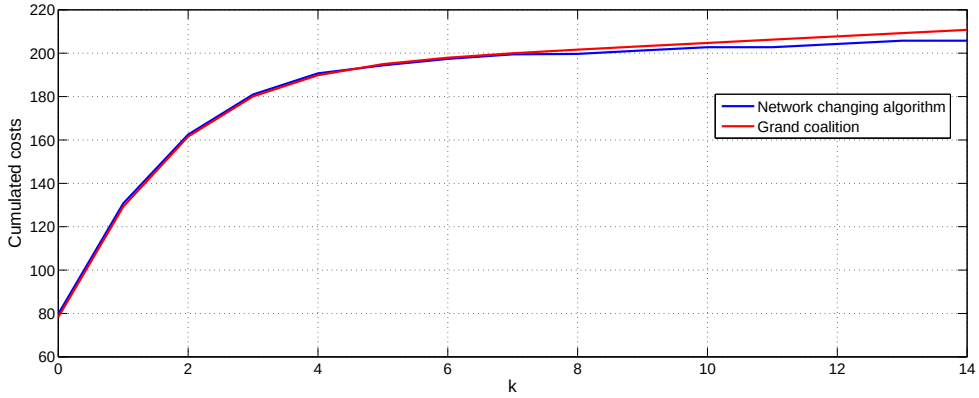


Figure 5.9: Cumulated cost.

the state is defined and they offer very useful and concentrated information through their respective Shapley values. It is possible to gain a valuable insight into the communication structure of the distributed system just with the information of these values. This information could be helpful for example to make an off line analysis about the relative importance of each of the links, so that useless links can be erased and the most important ones can be reinforced.

We present next some simulations of the proposed distributed controller which can be seen as a hierarchical control scheme. The highest level of hierarchy is executed every D seconds. In this level agents exchange their states and the current network mode is updated according to the state. The second level is executed every T seconds, with $T < D$. This level is responsible of implementing the corresponding control actions to the plant, so agents

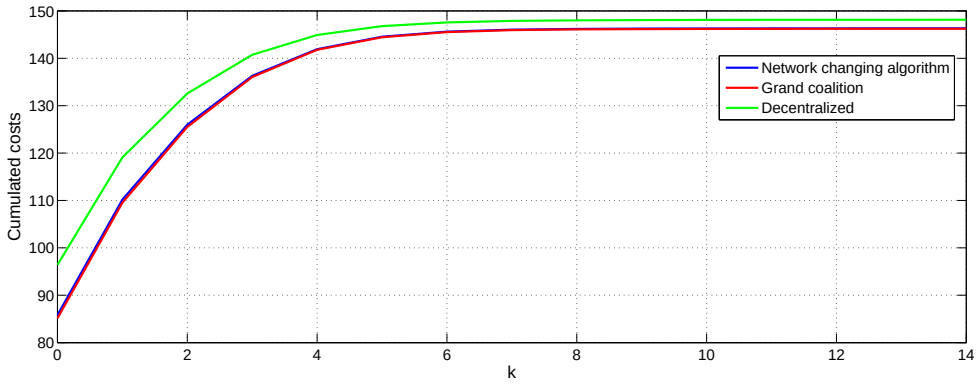


Figure 5.10: Cumulated cost without communication costs.

exchange their states according to the current network topology and the update the control action. The simulation presented here have been done with values of $D = 3$ and $T = 1$ and for the initial state:

$$x_1(0) = \begin{bmatrix} 2 \\ 1.8 \end{bmatrix} x_2(0) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} x_3(0) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} x_4(0) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Figures 5.6 and 5.7 show the evolution of the system states and inputs respectively as a function of time. Note that when one of the agents is not at the origin it disturbs the rest of the agents from their equilibrium point. Figure 5.8 shows the different network modes active during the simulation. In figure 5.9 the cumulated cost of the coalitional distributed algorithm is compared to the cumulated cost of applying full communication at each sample. Note that the additional communicational cost produced by the dynamic change of network mode is included in the cumulated cost of the coalitional distributed algorithm. For this reason it offers a higher cost during the first steps. Then, as the system is closer to the origin the advantages of the change of network mode can be seen. The advantages of the algorithm become even clearer when a comparison is made without taking into account the communication costs. In figure 5.10 the cumulated cost of the algorithm is compared to the cases of centralized and decentralized control. In this case, the cumulated cost of the proposed distributed controller is almost the same that the cumulated cost of the centralized controller, but this mode is hardly used as it is shown in figure 5.8. It can also be seen that decentralized control provides the worst closed-loop performance. Similar figures to the ones shown here have been obtained for other initial states. In each simulation, different modes come into play depending on which subsystems need the most to cooperate with their neighbors.

The techniques presented in the chapter can be applied to analyze WSAN systems in

which the number of agents is not high. The number of LMI's that have to be solved grows exponentially with the number of nodes as there are 2^N different communication nodes that have to be considered. However, this number can be reduced in the practice depending on the application. For example, not all the communication modes may make sense. In a wireless setting there are nodes that will never be able to communicate. Moreover, if the wireless nodes broadcasts their packages they may transmit at the same time to different receivers, which once more reduce the number of modes that have to be taken into account. In some cases, the designer may drop some of the communication modes because they are not probable. The on-line implementation of the algorithm may also be simplified. If only a change in one link is allowed every D sampling times, then the number of different modes that have to be explored is reduced to N. All these possible simplifications have to be studied for the particular problem considered.

5.4 Distributed estimation problem formulation

In the previous sections we have seen a scheme to dynamically manage the links of the network in a distributed control problem. In this section we focus on the dual of the distributed control problem, that is, the distributed estimation problem. As we pursue the same goals for this problem, we will transpose all the results developed for control to the estimation field. To this end, we consider the following uncertain distributed linear system

$$\begin{aligned} x_i(t+1) &= A_i x(t) + w_i(t) \\ y_j(t) &= \sum_{k \in I_j} C_{jk} x_k(t) + v_j(t) \end{aligned} \tag{5.13}$$

with $i \in N = \{1, \ldots, n\}$ and $j \in M = \{1, \ldots, m\}$, that is, there are n different subsystems whose states are given by $x_i(t) \in \mathbb{R}^{n_i}$ and m different outputs $y_j(t) \in \mathbb{R}^{q_j}$. The set $I_j \subseteq N$ stands for the set of states that contribute to the output j. The state and measurement noise components are characterized by normal distributions with zero mean and variances Q_i and R_j respectively; that is, $w_i(t) \in \mathbb{R}^{n_i}$ is a $\mathcal{N}(0, Q_i)$ and $v_j(t) \in \mathbb{R}^{q_i}$ is a $\mathcal{N}(0, R_j)$. Note that under this formulation it is not necessary to assign the outputs to any concrete subsystem.

The problem we face is to estimate the state of all the subsystems while minimizing a cost function that comprehends both estimation and communication costs. As we already know, the communication costs depend on the number of links that are used at each sampling time. The change of performance in the estimation will be analyzed in the case that some of the links were not *present* in the system. The analysis of the cost trade-off between estimation performance and communicational costs will determine what is the best network mode A at

time sample t.

Again, the network mode A is chosen every D sample times following a cooperative game approach and the agents are separated in groups, the so called communication components $C \subseteq N/A$. At each sample time, each communication component $C \subseteq N/A$ implements a Kalman filter using the available information; that is, the outputs and the current state estimation of all the agents that can be known inside C. From the point of view of an agent $i \in C$, we have the following filter

$$\hat{x}_i(t) = A_i\hat{x}_i(t-1) + \sum_{j \in Y_i^C} K_{ij}^A (y_j - \sum_{k \in I_j \bigcup \{i\}} C_{jk}x_k(t))$$

where Y_i^C is the set of outputs available in the communication component C that are completely determined by states of the agents inside this coalition and that offer information about the state of agent i. Mathematically, this set can be defined as

$$Y_i^C = \{j \in M | I_j \bigcup \{i\} \subseteq C\}$$

From the set of the matrices K_{ij}^A, the following observer is obtained for the centralized system characterized by the absence of communication for agents in different communication components of the network mode A

$$\hat{x}(t) = A\hat{x}(t-1) + K_A(y(t) - CA\hat{x}(t-1)) \tag{5.14}$$

where $x(t)$ and $y(t)$ are, respectively, the state and the output of the equivalent centralized system. The centralized state and observation matrices are $A = diag(A_i)$ and $C = [C_{ij}]$ for $i, j = 1, \ldots, N$. The matrix ${K_A}^2$ is the matrix which defines the centralized Kalman filter implemented for the network mode A. It is important to remark that the matrix K_A takes into account the communications constraints in A because it is made of the matrices K_{ij}^A.

In order to implement the proposed strategy we need to design the filter gain of (5.14) and provide a measure of the performance of each network mode. To this end we propose to use as a measure of performance the steady covariance matrix P_A to estimate at a given state the value of the uncertainty as

$$\phi_A(\hat{x}) = \hat{x}^T P_A^{-1} \hat{x}$$

Note that each network mode is characterized by a steady covariance matrix P_A, but in order to obtain a performance measure, the current estimate of the state $\hat{x}$ must be used. For this

[2]Note that in this section the matrix K_A stands for the Kalman filter gain while in the previous section it was used to denote the controller. We have not changed the notation to stand out the duality between the problems.

reason, this operation is done in a centralized manner; that is, when a new operation mode has to be chosen, all the agents must communicate their current state estimate.

In order to decide what communication strategy must be implemented, we use the quadratic function $\phi_A(\hat{x})$. This quadratic function will be used so decide the optimal communication mode as well as to define the link problem.

5.4.1 Network modes

In order to analyze which agents should communicate, the link game associated to the cost-extended communication situation H of the current estimate $\hat{x}(t)$ is studied. The characteristic function $r^H(A, \hat{x})$ assigned to each communication mode A is based on the error variance of the steady Kalman filter of all the communication components in the network defined by A. This function is defined as

$$r^H(A, \hat{x}) = \hat{x}^T P_A^{-1} \hat{x} + c|L(A)|, \ \forall A \subseteq L. \tag{5.15}$$

The best possible communication mode provides the winner network configuration choice. The function $r^H(A, \hat{x})$ will be also used to define the link game.

Note that, again, we can speak of regions of dominance. We will denote CR_A as the set of points for which a network mode A is dominant is characterized by the states such that A provides the best (lower) cost.

5.4.2 Link analysis

The link game is constructed by a set of players composed by the links of the network L and a characteristic function that assigns to each communication mode A an given utility. In this case, we propose to use $r^H(A, \hat{x})$ to define the link game. Once the link game is constructed, a qualitative and quantitative analysis from the importance of the links in the game may be obtained from the corresponding Shapley value $\gamma(L, r^H, \hat{x})$. Each component of this vector represents the cost of a given link for the system. In other words, the lower value a link has, the higher utility it has for the system.

5.4.3 Agent analysis

A qualitative and quantitative analysis from the importance of the agents of the system may be obtained from the Shapley value $\gamma(N, w^H, x)$ of the corresponding agent game. To build such game it is necessary to define the characteristic function that assigns a value to each coalition S of players for a given network L. First we define the cost of a communication component

$$\phi_C(t, \hat{x}_C) = \hat{x}_C(t)^T P_C^{-1} \hat{x}_C(t) + \gamma c|L(C)| \tag{5.16}$$

where $\hat{x}_C(t)$ is obtained from the current state estimate $\hat{x}$ and P_C is the steady covariance matrix of the communication component C assuming that there is no communication with agents outside the coalition.

Based on these functions, the characteristic function of the agent game that defines the utility of a coalition S is defined as

$$\omega^H(S, x) = \sum_{C \in S/L} \hat{x}_C(t)^T P_C^{-1} \hat{x}_C(t) + \gamma c|L(S)|, \ \forall S \subseteq N.$$

The Shapley value of the game (N, v^H, x) provides concise information about the relevance of all the players in the game. The lower the value the is, the more relevant role the agent has in the game. It is important to stand out that the Shapley value of the agent game, as it is defined, does not have any physical meaning.

5.4.4 Coalitional game design method

In this section we present a method to design for a given system all the matrices that define the Kalman gain matrices for for both each of the networks modes (K_A and P_A for all $A \subseteq L$) as well as each of the possible communication components of the agent problem (K_C and P_C for all $C \subseteq N/L$).

In what it follows it is assumed that the centralized system matrices A and C are constant and observable. The covariance matrices Q and R are also assumed to be constant and known. Under these assumptions it is possible to calculate offline the matrices needed to implement the Kalman filter.

First, we deal with the problem of finding the matrices that define the estimators and the weights of the cost functions for both each of the possible communication components of the agent problem (K_C and P_C for all $C \subseteq N/L$). In this case, K_C must estimate the states of C assuming that the outputs that do not belong to that communication component are zero.

The following theorem, which is a particularization of the theorem 1 in [7], provides an LMI constraint that can be used to design appropriate matrices.

Theorem 9 *Let $C \in S/L$ be a set of independent communication components for a given communication situation H whose dynamics are given by $A_C = diag(A_i)$, $\forall i \in C$ and $C_C = [C_{ij}]$, $\forall i, j \in C$, and let the noise be characterized by $Q_C = diag(Q_i)$ and $R_C = diag(R_i)$, $\forall i \in C$. If there exist matrices W_C and S_C such that the following optimization problem*

$$\begin{array}{l} \max tr(W_C) \\ s.t. \\ \left[\begin{array}{cccc} -W_C & W_C A_C - S_C C_C A_C & W_C - S_C C_C & S_C \\ * & -W_C & 0 & 0 \\ * & * & -Q_C^{-1} & 0 \\ * & * & * & -R_C^{-1} \end{array}\right] < 0 \end{array} \tag{5.17}$$

then the filter gain and the error variance of the steady Kalman filter are $K_C = (W_C)^{-1} S_C$ and $P_C = (W_C)^{-1}$.

In the agent game the approximation of the cost of the communication component C is based on the cost given by $\hat{x}_C^T P_C \hat{x}_C$. This upper bound is calculated assuming that the rest of the agents states are null. Note the agent game only provides grounds for distributing the benefits or costs between the agents during the game.

The link game imposes a different perspective for the communication situation. In this case the grand coalition controls the system having into account all the possible network configurations $A \subseteq L$. Each A imposes communicational constraints and divides the system in a set of communication components $C \in N/A$. Following the same approach as in the agent game, the following theorem is presented.

Theorem 10 *Let $A \in L$ be a set of active links for a given communication situation H. The dynamics of the whole system are given by $A = diag(A_i)$, $\forall i \in N$ and $C = [C_{ij}]$, $\forall i, j \in J$ and its noised matrices defined by $Q = diag(Q_i)$ and $R = diag(R_i)$, $\forall i \in J$. If there exist matrices $W = [W_{ij}]$, $\forall i, j \in N$, where $W_{i,j} \in \mathbb{R}^{n_i \times n_j}$, and $S = [S_{ij}]$, $\forall i, j \in N$, where $S_{i,j} \in \mathbb{R}^{n_i \times q_j}$. If there exist matrices W and S such that the following optimization problem*

$$\begin{array}{l} \max tr(W) \\ s.t. \\ \left[\begin{array}{cccc} -W & WA - SCA & W - SC & S \\ * & -W & 0 & 0 \\ * & * & -Q^{-1} & 0 \\ * & * & * & -R^{-1} \end{array}\right] < 0 \end{array} \tag{5.18}$$

then the filter gain and the error variance of the steady Kalman filter are $K = (W)^{-1}S$ *and* $P = (W)^{-1}$.

5.5 Distributed estimation simulation results

This section presents an example to show the strongest result of this work: the dynamical change of the communication mode in a network. The strategy we propose is based on the simulation scenario proposed for the localization of robots in [31] that was also used in chapter 4.

Let us consider a system consisting of a set $\mu = \{1, \ldots, M\}$ of reference nodes or beacons and a set $\eta = \{1, \ldots, J\}$ of mobile devices. In this example we will consider $M = 6$ beacons and $J = 8$ mobile devices, which are located in the positions depicted in figure 5.11.

The goal is to estimate the position of the moving devices. We assume that the sample time is sufficiently small to consider that the devices' displacement at each sample is small enough to be considered noise. The equations for each device are:

$$x_i(t+1) = x_i(t) + \Delta x_i(t) \; \forall i \in \eta = \{1, .., J\}$$

where $x_i(t) \in R^2$ is the position of the i-th robot at time t and $\Delta x_i t \in R^2$ with $x_i(0) = x_i^0$. The beacon position is fixed so that $x_i(K+1) = x_i(0) \; \forall i \in \mu = \{1, .., M\}$. The distance between the nodes and the mobile devices can be calculated using

$$d_{ij}^2(t) = (x_i(t) - x_j(t))^T (x_i(t) - x_j(t)) \; \forall i, j \in \eta, \mu.$$

The distance can be linearized around the steady state positions $\overline{x}_i$ using a first order Taylor approximation, which leads to

$$d_{ij}^2 = \overline{d}_{ij}^2 + 2(\overline{x}_i - \overline{x}_j)^T (x_i - \overline{x}_j) + 2(\overline{x}_i - \overline{x}_j)^T (\overline{x}_i - x_j)$$

with $\overline{d}_{ij}^2 = d_{ij}^2(\overline{x}_i, \overline{x}_j)$. Now, system variables can be introduced for all the mobile devices such that:

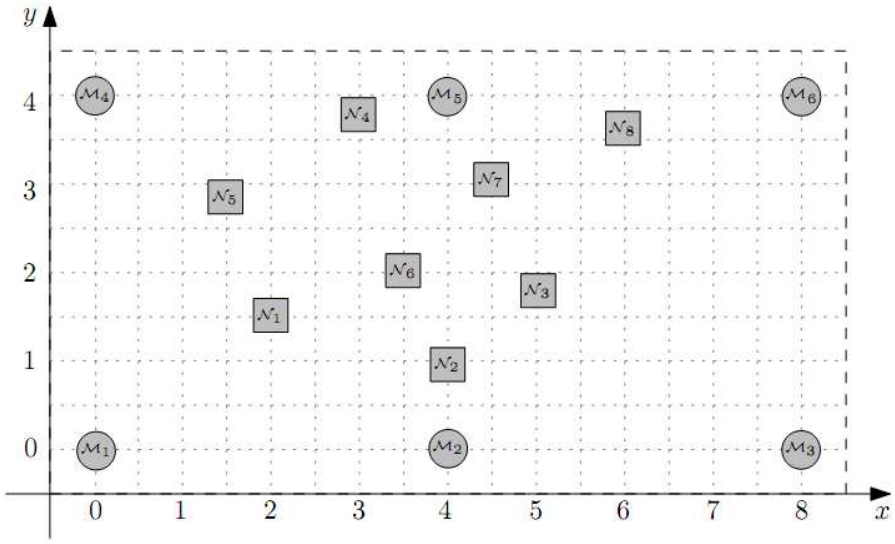


Figure 5.11: Initial situation of the devices.

$$
\begin{aligned}
&x_i(t) = x_i(t) - \overline{x}_i \ \forall i \in \eta \\
&y_{ji}(t) = d_{ij}^2 - \overline{d}_{ij}^2 \ \forall i \in \eta, \forall j \in \eta, \mu \\
&C_{ji} = 2(\overline{x}_i - \overline{x}_j) \ \forall i \in \eta, \forall j \in \eta, \mu.
\end{aligned}
$$

Each moving device's output provides information of the distance with respect the other moving devices and the beacons. If white gaussian additive noise is assumed in the state and output then each device can be modeled according to equation (5.13).

In order to make the situation more realistic we assume that only devices and beacons within a given range can communicate. Thus a communication radius ρ is defined. In general two devices i and j can communicate if $d_{ij} < \rho$. A communication graph can be defined to reflect which devices can communicate at each sample time. The initial communication graph for the employed value of $\rho = 2.5$ is given by the following matrices:

$$A_0^{\eta} = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

$$A_0^{\mu} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

where $A_0^{\eta}(i,j) = 1$ if the mobile device i is able to communicate with the mobile device j and $A_0^{\mu}(i,j) = 1$ if the mobile robot i is able to communicate with the beacon j. As long the devices move the communication graph will change with the time.

If we assume that each moving devices is able to establish individual communication with the rest of the devices, then it is straight forward to check that the number of possible links ascends to $(J^2 - J)/J$. In this case we have a total amount of 28 possible links. As any link can be either active or inactive then it is possible to define 2^8 possible network modes. However, given that only 16 links are available due to the range constraint, the number of network modes that have to be compared is reduced to 2^{16}. In practice the number of modes to be compared is much lower due to the fact that many of the modes are redundant because they connect the same set of devices. Moreover, there are modes that are not redundant but make no sense because they are based on long routes to connect the devices. An offline analysis is recommended so that the on-line burden is reduced to the comparison of a lower number of modes. In addition, it is recommended to use a link cost that penalizes the distance between nodes as a way to obtain those modes with shorter links during the offline analysis.

The noise considered for the design of the Kalman gains was defined by $Q_i = diag(0.001)$ and $R_i = 0.05$ for all $i \in [1, J]$ and the weighted link unitary cost was $c = 0.1$ (this value was obtained after a proper tuning procedure). However, in order to test the robustness of the proposed distributed estimation scheme, the simulations were done with a noise higher than

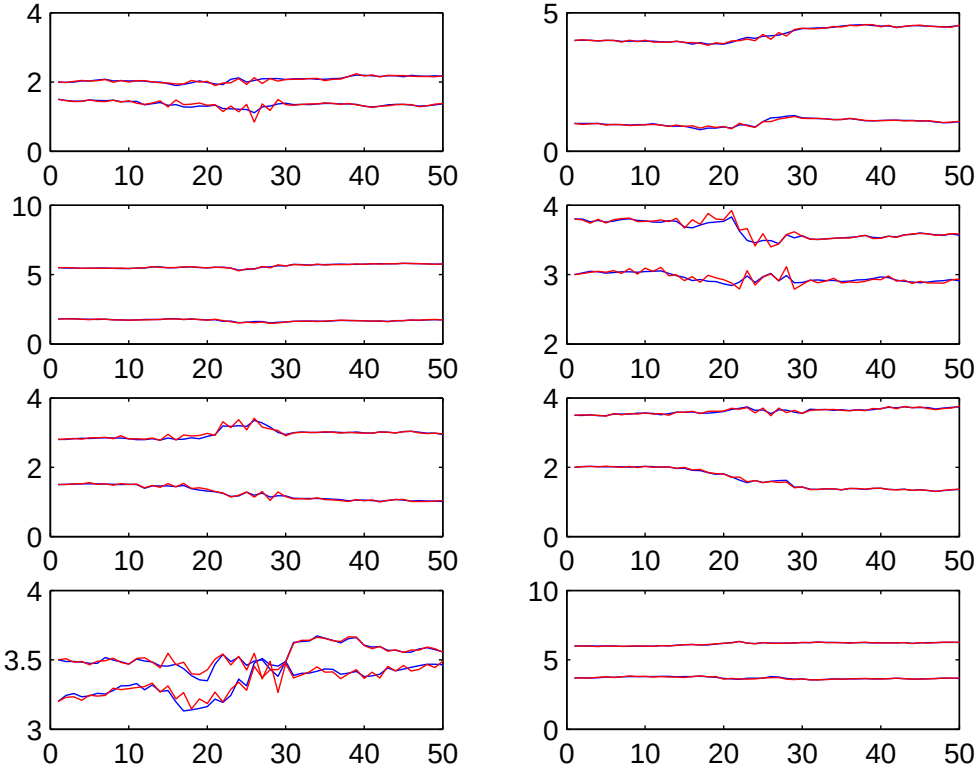


Figure 5.12: Agents state evolution.

the employed in the design procedure. The system has been simulated for 50 time samples. Each 10 samples it was decided what was the best mode to continue the state estimation. During the first 10 time samples the state noise was 20 times stronger. During the next 10 samples this value was 40. Then it was incremented again up to 80 during 10 more samples. Finally and until the rest of the simulation it was reduced to 20 again.

In figure 5.12 it can be seen the time evolution of the states during the simulation. In blue it is depicted the actual state and in red it can be seen the estimation. In figure 5.13 it can be seen the trajectory of the plane of the robots. Again in blue it is the real position and in red it can be seen the estimated.

During this simulation the following modes were decided by the distributed control mechanism. During the first 20 samples the system worked decentralized, which is natural given that agents begin at the linearization point of the system. After that some links are enabled to cope with the increasing deviation from the linearization point. In particular the

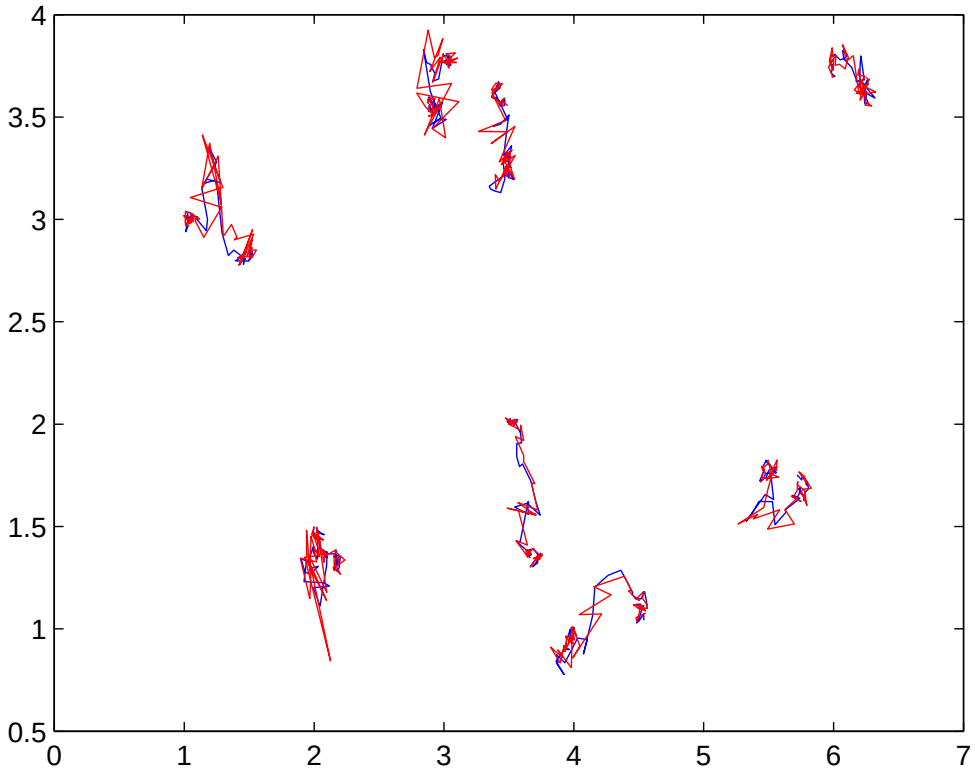


Figure 5.13: Agents state plane trajectory.

links between the pair of agents (2,6), (3,6), (5,6) and (3,8) are enabled, which means that agents 2, 6, 3, 8 and 5 are in the same communication component while the others work decentralizedly. Finally, after 10 samples and until the end of the simulation, the distributed estimation scheme implements the grand coalition to estimate the state of the system, that is, 7 links are on so that all the agents can communicate using the network. We have considered the same cost c for all the links in the system. However if the link distance is taken into account then the shortest path that communicates all the agents will be implemented when the great coalition is implemented.

5.6 Conclusions

This chapter provides a bridge between coalitional game theory and control. In particular, an application of cooperative game theory to analyze distributed control and estimation schemes has been proposed. The modeling of distributed systems from the game theory point of view allows us to extract useful information about the communication structure of a system and the relative importance of the agents and links. It is possible to use this information for interesting applications such as the online change of network mode to optimize the use of the communication resources. The main contribution of the chapter from the control point of view consists on the dynamic switching of the communication links as a part of the control algorithm. In addition, an optimization based design method has been provided for the class of systems considered.

Chapter 6

Conclusions and Future Research

In this thesis we have developed new distributed control and estimation schemes based on unifying model predictive control and game theory. As it has been seen, game theory provides an appropriate framework to tackle the class of problems that appear in distributed architectures. Moreover, game theory allows one to obtain a deeper insight in distributed problems in comparison with other approaches which consist on a mere distribution of the calculations needed to solve the centralized optimization problem.

It is worthwhile to stand out that in this thesis a great effort has been made obtain control and estimation schemes with a low communicational burden. Likewise, we have developed techniques to switch dynamically the links of a network with the goal of saving unnecessary communications. In general, previous distributed algorithms have been more focused on reducing the computational burden instead. This implies that many distributed schemes are not suitable for systems in which there are communicational constraints. On the other hand, there has been a price to pay in terms of performance due to the simplifications that were made in order to reduce the communicational complexity of our distributed solutions. Nevertheless, our results show that our schemes provide a very good trade-off between performance and communicational burden.

One of our most important objectives during this work has been to minimize the amount of information about the centralized model that the agents need to have in order to implement the proposed schemes. In particular we have focused on schemes that do not require the agents to share information about their state and objectives. Actually, in our framework the only information about the rest of the system that an agent has is the way its neighbors affect it, which from our point of view a very reasonable assumption.

Finally, it is important to stand out that all the schemes and techniques developed throughout this work have been tested, at least, in simulation.

6.1 Conclusions

We present next the main contributions of each of the chapters of this thesis:

- **Distributed Model Predictive Control and Game Theory**. In chapter 1 we have explained the main problems associated to distributed model predictive control. Basic concepts and taxonomies for both game theory and distributed control were given. In addition, a profound literature review of previous distributed MPC results has been done from the communicational point of view, that is, special attention has been paid to the type and amount of information shared by the agents.

- **Distributed Model Predictive Control Based on Game Theory for Two Agents**. In chapter 2 we focused on distributed model predictive control for systems controlled by two agents. A novel algorithm with low communicational requirements based on game theory was proposed and put to test with simulated and real examples. It is also important to remark that each agent solves an optimization problem that only depends on its local model and partial state information. For this reason, the algorithm is suboptimal since the agents have an incomplete view of the system and propose the best solutions from their point of view. In addition, we have provided sufficient conditions that guarantee practical stability of the closed-loop system as well as an optimization based procedure to design the controller so that these conditions are satisfied.

- **Distributed Model Predictive Control Based on Game Theory for Multiple Agents**. In chapter 3 the ideas proposed in chapter 3 were extended for the general case of a system controlled by more than two agents. The original algorithm proposed in chapter 2 could not be readily extended due to the combinatorial explosion of possible strategies that appear in a multiple agent problem. In this case the agents make proposals to improve an initial feasible solution on behalf of their local cost function, state and model at each sample time. These proposals are only accepted if the global cost improves the cost corresponding to the current solution. The agents exchange a greater number of suboptimal proposals in comparison with the algorithm presented in chapter 2 but our simulations showed that still a good performance can be achieved with a low number of communications per agent. The proposed algorithm provides a feasible solution to the centralized problem. Finally, we introduced a new concept of invariance for distributed and decentralized systems that guarantee that the closed-loop

system is practically stable along with an optimization based controller and invariant design procedure.

- **Distributed Receding Horizon Kalman Filter**. In chapter 4 a distributed version of the Kalman filter based on dynamic programming was developed. In this case, the distribution of the centralized problem between the agents was done by means of dual decomposition, which is one of the most popular techniques for distributed control problems. It is common that in dual decomposition agents exchange their state in order to update the prices introduced by the Lagrange multipliers. For this reason, different coordination alternatives for the price update were considered so that agents do not need to exchange their state. Note that this feature may be interesting in control applications in which dual decomposition is used as well. The techniques developed in this chapter were tested for a simulated application for the self-localization of robots.

- **Applications of Cooperative Game Theory to the Control and Estimation of Distributed Systems**. In chapter 5 a bridge between coalitional game theory and control has been built. The main result of this chapter was a distributed scheme that dynamically enabled or disabled links of a network with the goal of saving communications. In this sense, an optimization based controller and estimator design method was provided for the class of systems considered. In addition, it was shown how to use the Shapley value in order to extract useful information about the communication structure of a system and the relative importance of the agents and links.

6.2 Future research

Although the research of distributed systems have become a hot issue in the last decade, there are still many interesting topics that will have to be studied in the future. In this section we enumerate some research lines that are interesting from our point of view:

- Many distributed schemes have been proposed but few have been tested in real systems. Real applications demand communication protocols specially designed for distributed control. In addition, it has to be studied how distributed techniques can be applied in low resource systems. Home networking technologies point towards the ambient intelligence paradigm, a scenario where the cooperation among the electronic systems at home is essential. This topic constitutes an interesting research line in telecommunications engineering.

- Real communication networks do not behave ideally. In 1992, Peter Deutsch provided a list with eight typical assumptions usually made when building distributed applica-

tions [21]. All of them prove to be false in the long run and therefore cause "painful learning experiences". The complete list of fallacies of distributed computing is:

1. The network is reliable.
2. Latency is zero.
3. Bandwidth is infinite.
4. The network is secure.
5. Topology doesn't change.
6. There is one administrator.
7. Transport cost is zero.
8. The network is homogeneous.

As it can be seen, some of this fallacies have been addressed in this thesis, specially those related with the transport cost and reliability. In the literature there are several works that deal with the dynamics induced by the communication network (such as time-varying delays and data losses) [63, 20] but very few has been investigated about topology changing networks. The homogeneity and security of the network are also common assumptions many works. Probably, the main reason for this is that by now distributed applications are developed *ad hoc* for certain systems. However, these issues will have to be addressed in the future before the application of distributed techniques is successful in real world applications.

- Although game theory is often applied to explain human behavior, it is the field of automatic control where its assumptions of players' rationality and intelligence hold better. According to [96], the "hyper-rationality" of game theory may actually be an appropriate model for software agents. For this reason, it will be necessary to apply game theory tools to prevent selfish agents mechanisms to take advantage of distributed applications. In this sense, the application of mechanism design may be interesting. Mechanism design is a branch of game theory that attempts to implement desired social choices in a strategic setting. In this context, a social choice is defined as an aggregation of the preferences of the different participants toward a single joint decision. Examples of social choices are: elections, markets, auctions... Deep down, any cooperative distributed control scheme is a way to determine a social choice. Therefore, it would be interesting to develop techniques to so that distributed algorithms are prepared in case the agents show strategic selfish behavior.

- It will be important for future works to provide some kind of suboptimality bounds to determine the performance obtained by distributed estimation schemes. In the case of distributed control interesting results can be found in [28]. However, the application of the results of this paper to the estimation problem is not straight forward. Sub-optimality bounds are important since they allow to establish the trade off between performance and communicational burden in an explicit way.

- Finally, it would be desirable to transpose results from welfare economics into distributed control and estimation problems. Under several assumptions, in welfare economics it is shown how selfish agents may achieve results optimal in the sense of Pareto behaving selfishly. This approach is based mainly in the idea of exchanges that have positive surplus for all the agents involved in the transactions. Note that, in a certain way, this is the same idea that is behind the control scheme proposed based on agent negotiation (chapter 3). Nevertheless, it would be interesting to investigate how to modify the scheme in order to determine under which conditions centralized optimality could be achieved.

Appendix A

Resumen en castellano

Desde hace miles de años el hombre ha soñado con aparatos que funcionen solos. Homero, por ejemplo, hace referencia en la Ilíada a unos trípodes fabricados por Hefesto con ruedas de oro en los pies "para que del propio impulso pudieran entrar donde los dioses se congregaban y volver a la casa", acción ésta que el propio texto calificaba como "¡Cosa admirable!" y que refleja la emoción que la propia idea suscita. Precisamente, este es el objeto de la teoría de control autómatico, que es la rama de la ingeniería que comprende aquellos conocimientos técnicos necesarios para hacer que las cosas funcionen por sí mismas.

En general, el primer problema que el ingeniero de control debe resolver es el de obtener un modelo matemático que sintetice el conocimiento previo que se tiene sobre el comportamiento del sistema u objeto que se pretende controlar. Dicho modelo proporciona información de tipo causa-efecto y permite calcular qué acciones de control deben llevarse a cabo para que el sistema se comporte de la manera deseada. El propio ser humano se basa en estos mismos principios cuando *controla*, es decir, cuando ejerce acciones sobre un objeto encaminadas a obtener un cierto resultado. Por ejemplo, imaginemos a una persona al volante de un vehículo. Es evidente que el conductor decide qué acciones ha de realizar (girar el volante, acelerar, frenar...) a partir de la información disponible sobre el estado del coche (posición en la carretera, velocidad,...) y del modelo mental que tiene sobre el comportamiento del coche.

El ejemplo anterior permite señalar dos elementos fundamentales de cualquier problema de control: la información y el tiempo. La información disponible determina la *calidad* del control que puede llevarse a cabo. Cuanto mejores sean el modelo del sistema y las medidas del estado, mejores decisiones podrán tomarse. Por su parte, el tiempo para la toma de decisiones no es infinito. La dinámica del sistema impone restricciones temporales para la toma de decisiones que no deben violarse; una buena acción de control aplicada de forma

tardía bien puede convertirse en una mala acción de control. En el ejemplo del coche queda claro que el riesgo de que el conductor falle en el control del vehículo es mucho mayor si éste conduce con los ojos tapados (menos información) o a 250 km/h (tiempo para la toma de decisiones reducido).

Tradicionalmente la teoría de control se ha enfrentado a las limitaciones impuestas por la información y el tiempo de una forma centralizada. En otras palabras, el diseño de los dispositivos de control se realiza suponiendo que se dispone de toda la información necesaria en un único punto donde ha de tomarse una decisión en el tiempo preciso. Es indudable que esta forma de proceder proporciona los mejores resultados posibles. Por desgracia, no siempre se puede trabajar de manera centralizada. Hay diferentes razones que lo impiden. Entre ellas destacan las siguientes:

- La complejidad del sistema es tal que es imposible obtener un modelo que determine su comportamiento globalmente.
- Aunque pueda obtenerse un modelo, éste es tan complejo que no es posible procesar toda la información necesaria para la toma de decisiones en un tiempo razonable.
- El sistema se extiende en un área lo suficientemente grande como para que no sea posible concentrar la información de las medidas en único punto en un tiempo razonable.
- El sistema está compuesto por diferentes entidades que interactúan entre sí y alguna de ellas no quiere revelar su modelo de funcionamiento.

Cuando se presenta alguna de las situaciones anteriores, no se puede encontrar una solución centralizada al problema de control. Es en este punto en el que entran en juego los sistemas de control descentralizados y distribuidos. La filosofía de estos esquemas es sencilla: el problema de control global se divide en varias partes diferentes, cada una las cuales es asignada a un controlador local o *agente*. Por lo tanto, cada agente carece de una visión de conjunto del problema centralizado, lo cual constituye la caracterísitica más importante de este tipo de sistemas. En función del grado de interacción existente entre las diferentes partes en el que se divide problema centralizado, es posible que deban establecerse mecanismos de comunicación entre los agentes para que trabajen de forma coordinada. De esta manera, cuando el grado de interacción es bajo, los agentes pueden trabajar sin comunicarse entre sí, por lo que se habla de sistemas descentralizados. En cambio, cuando se dispone de medios para que los agentes trabajen coordinadamente se habla de sistemas distribuidos.

En contra de lo que pueda parecer, trabajar de forma decentralizada o distribuida también presenta importantes ventajas, algunas de las cuales justifican por sí solas que se recurra a este tipo de esquemas aun cuando no resulte estrictamente necesario. Algunas de estas ventajas son:

- Simplicidad de las soluciones: en lugar de resolver un problema difícil, se resuelven varios problemas más sencillos.
- Escalabilidad: resulta muy sencillo ampliar un sistema distribuido dada su modularidad inherente.
- Facilidad de mantenimiento: no hay que detener el proceso controlado cada vez que se llevan a cabo tareas de mantenimiento en el sistema de control.
- Robustez: al no depender del correcto funcionamiento de un único elemento, el sistema de control resulta más tolerante a los fallos que puedan presentarse.

Como puede suponerse, disfrutar de estas ventajas tiene un precio expresable en términos de pérdida de rendimiento frente al control centralizado. El grado de pérdida depende del nivel de interacción entre las diferentes partes del problema de control centralizado y de los mecanismos de coordinación existentes entre los agentes. En general, la calidad de las decisiones de control dependerá del intercambio que se establezca entre el número de comunicaciones que los agentes realizan para coordinarse y los costes derivados del hecho de la comunicación, como son el tiempo empleado, la carga computacional o el consumo eléctrico de los dispositivos, factor crucial en muchas aplicaciones. Es justamente en este punto en el que se desarrolla la presente tesis doctoral, que tiene por objeto desarrollar técnicas de control distribuido que ofrezcan un buen desempeño al mismo tiempo que minimicen el coste comunicacional.

A.1 Redes de sensores y actuadores

El concepto de sistema descentralizado o distribuido no es nuevo, sin embargo no ha sido hasta los últimos años cuando se ha producido el verdadero auge de las líneas de insvestigación de control distribuido. El interés por este tipo de sistemas ha venido motivado por el advenimiento de los transceptores inalámbricos de bajo coste y su aplicación a las redes de sensores y actuadores.

Las redes de sensores y actuadores inalámbricas constituyen una de las áreas más importantes dentro de la ingeniería de control en la actualidad. Cualquier sistema, sin importar su naturaleza, posee una serie de magnitudes susceptibles de ser medidas o influenciadas, por lo que cualquier tecnología que incida en estos puntos está directamente relacionada con las propias raíces del control automático. Hasta no hace muchos años, la única manera de llegar a los puntos de medición o actuación era a través de cables, lo que implica el despliegue de una infraestructura relativamente costosa. Los primeros sistemas inalámbricos existentes

proporcionaban una solución a este problema, si bien su coste y su elevado consumo eléctrico impedidieron que se generalizara su uso. Nótese que un consumo elevado de electricidad implica o bien una renovación muy frecuente de las baterías, o bien el despliegue de cables que alimenten a los dispositivos.

Este panorama cambió con la llegada de tecnologías basadas en el estándar IEEE 802.15.4, como por ejemplo Zigbee. A diferencia de otras tecnologías que, como Wifi o Bluetooth, están orientadas a la provisión de un elevado ancho de banda, Zigbee se concentra en la reducción del consumo eléctrico de los dispositivos, hasta el punto de que pueden pasar años sin necesidad de cambiar las baterías. Por ello, ofrece un ancho de banda relativamente bajo, aunque más que suficiente para la información que se necesita transmitir en una gran cantidad de aplicaciones de control. Este factor, unido con la rapidez de despliegue de las redes inalámbricas, ha provocado una auténtica explosión en las aplicaciones de esta tecnología y, por tanto, de los sistemas distribuidos.

En la actualidad se viven tiempos emocionantes para el ingeniero de control gracias a las redes de sensores y actuadores inalámbricas. Sin duda se trata de una tecnología que revolucionará el mundo del control tal y como lo conocemos, tanto a nivel doméstico como industrial. Se abren multitud de interrogantes que deben ser debidamente estudiados para aprovechar de forma óptima las nuevas posibilidades a nuestro alcance. Por ejemplo, paradigmas como el de la sensorización ubicua eran poco más que una utopía hasta hace poco tiempo. Ahora que la tecnología permite convertir en realidad este antiguo sueño, la comunidad de control tiene ante sí un fértil campo de estudio que habrá que saber sembrar adecuadamente para recoger resultados a la altura de las expectativas creadas.

En este orden de cosas, cobran importancia capital líneas de investigación como son el análisis y estudio de sistemas descentralizados, la realización e implementación de algoritmos de control distribuidos, la integración de diferentes tecnologías dentro en una misma red o el aprovechamiento óptimo de la información manejada por la misma, por citar solo algunos de los campos de investigación en cuya intersección puede encuadrarse este trabajo. Estamos pues ante una tarea eminentemente pluridisplicinar, cuyo desarrollo exige la integración de resultados provenientes de ramas muy diversas.

A.2 La teoría de los juegos

Mención especial merece la teoría de los juegos en esta tesis, una joven pero prolífica disciplina de las matemáticas cuyo origen puede situarse en 1944 con la publicación del libro *Game Theory and Economic Behaviour* de Von Neumann y Morgenstern [100]. La teoría de los juegos estudia situaciones o *juegos* en las que una serie de agentes o jugadores con poder de

decisión intentan conseguir unos determinados objetivos que pueden estar en conflicto entre sí. Todos los jugadores son conscientes de que el resultado final depende, no sólo de sus decisiones individuales, sino de las decisiones tomadas por el resto de jugadores. La teoría de juegos, por tanto, se centra en el estudio de lo que podríamos denominar como situaciones de decisión interactiva. Esta interactividad sitúa a la teoría de juegos en clara contraposición con los problemas de optimización habituales en ingeniería, en los que hay un único ente con poder de decisión.

A pesar de que la teoría de juegos ha sido aplicada con gran frecuencia para estudiar situaciones económicas de todo tipo, existen otros muchos campos donde su aplicación es directa como son la biología, la sociología, las ciencias políticas o las propias ingenierías. Habida cuenta de que en este trabajo de investigación se trabaja con sistemas distribuidos, es natural que se produzcan situaciones en las que aparece la interacción que estudia la teoría de los juegos. Por consiguiente, dicha teoría tiene un papel protagonista en esta tesis.

En general, cada controlador en un sistema distribuido se encarga de un subproblema del problema original. La probabilidad de que las decisiones individuales de cada uno de los controladores coincidan con las que conducen a un resultado óptimo desde el punto de vista global es remota, especialmente cuando los intereses individuales de los diferentes controladores entran en conflicto entre sí. La teoría de los juegos nos enseña que la interacción entre los controladores llevará al sistema a una de las siguientes situaciones:

- Inestabilidad. Se produce cuando se forma un ciclo pernicioso entre las acciones de unos agentes y las reacciones de otros. El bucle entre acciones de unos y respuestas de otros acaba conduciendo a un comportamiento nocivo para todos los agentes implicados.

- Equilibrio de Nash. En caso de que cada agente esté satisfecho con sus decisiones después de haber observado las decisiones de los demás, no habrá incentivos para que ningún agente cambie su línea de actuación en el futuro. La estabilidad de este tipo de equilibrios tiene un precio en términos de rendimiento o coste del control, por lo que en principio no son muy deseables. No obstante, la estabilidad es una característica tan deseable que hay esquemas de control distribuidos que se basan en el equilibrio de Nash [43],

- Óptimo de Pareto. Se dice que se ha alcanzado un óptimo de Pareto cuando las acciones realizadas por los agentes son tales que no existe la posibilidad de que ningún agente mejore sus resultados sin empeorar los de otro. La solución óptima del problema es en sí misma un óptimo de Pareto, el mejor que puede conseguirse. Idealmente, los agentes deberían mostrarse predispuestos a colaborar entre sí con el objeto de encontrar algún tipo de óptimo de Pareto, aunque la predisposición y coordinación necesarias no son siempre posibles.

De las tres situaciones anteriores, es evidente que la primera resulta totalmente indeseable, lo que nos permite obtener una conclusión muy valiosa para el control de sistemas distribuidos: el primer objetivo de todos los agentes de un sistema de control distribuido es garantizar la estabilidad del sistema. Una vez conseguido ese objetivo, cada agente puede preocuparse de perseguir sus objetivos individuales. En este punto conviene destacar que, lamentablemente, la estabilidad de un sistema de control distribuido no puede garantizarse sin un análisis desde el punto de vista centralizado del mismo. A pesar de lo paradójico de la última afirmación, resulta razonable que no se pueda certificar una propiedad global de un sistema sin conocer en detalle el comportamiento global del mismo. Por otra parte lo contrario no tiene por qué ser cierto, es decir, el desconocimiento del modelo centralizado de un sistema distribuido no implica la inestabilidad del sistema; lo único que implica es que no puede garantizarse la estabilidad del sistema a priori.

Todas las dificultades provenientes de la interacción entre las decisiones de un conjunto de agentes giran en torno a un concepto muy sencillo: la empatía. En el lenguaje cotidiano la empatía se define como la identificación mental y afectiva de un sujeto con el estado de ánimo de otro. Este concepto debe interpretarse en un sentido más amplio en este contexto; si cada agente conoce y valora el impacto de sus acciones sobre el resto, es más fácil que se llegue a una situación de equilibrio para todas las partes. La anterior afirmación descansa explícitamente en dos suposiciones:

- Cada agente *conoce* el impacto de sus acciones sobre el resto, lo que implica que o bien cada agente dispone de la suficiente información del problema global como para conocer las consecuencias de sus acciones, o bien los agentes disponen de algún mecanismo de comunicación que les permita transmitirse entre ellos el impacto que las acciones de cada uno tiene sobre el resto.

- Cada agente *valora* el impacto de sus acciones sobre el resto, lo que implica que cada agente hace suyo, hasta un cierto punto al menos, el *bienestar* del resto. Por lo tanto, la información sobre el impacto de un agente sobre el resto se utiliza con un fin social y no individual.

En caso de que ninguna de estas suposiciones se cumpla es complicado, cuando no imposible, que el resultado de las interacciones del juego sea favorable para los intereses de todos los jugadores. En esta tesis se parte de la base de que las dos suposiciones anteriores se mantienen, algo nada descabellado en el control de sistemas distribuidos. En concreto se utiliza la red de comunicación que une a los agentes para que éstos compartan información acerca del impacto de las acciones de unos sobre otros. De este modo, ningún agente necesita conocer detalles de la dinámica del resto, hecho éste que refuerza el carácter distribuido de la solución a la vez que aleja este trabajo de otros esquemas existentes en la literatura [99].

Por otra parte, la necesidad de una solución que dé estabilidad al sistema en bucle cerrado, exige que cada agente deba preocuparse por el *bienestar* del resto.

A.3 Objetivos de la tesis

Como se verá a lo largo de la tesis, las técnicas de control distribuido que se han desarrollado buscan resultados que conduzcan al óptimo de Pareto. Por desgracia, y tal y como ya se ha dicho, la búsqueda de este tipo de soluciones es compleja y en general requiere un uso intensivo de la red de comunicación. En los últimos años se han desarrollado soluciones de control predictivo distribuido comunicacionalmente intensas que abordan de el problema de esta forma [71, 99]. El enfoque por el que se ha optado en la tesis simplifica el problema de alguna de las maneras siguientes con objeto de ahorrar comunicaciones:

- Reducción del abanico de acciones que cada agente puede ejecutar. Dicha reducción puede ir encaminada a limitar las perturbaciones que los agentes inducen sobre el resto del sismema o a disminuir que el número de opciones a considerar. El precio de esta medida es expresable en términos de optimalidad de la solución de control, que al surgir de un universo de opciones más pequeño posiblemente ya no sea óptima desde un punto de vista centralizado. No obstante, si el ahorro comunicacional es importante y la pérdida de rendimiento en el control es pequeña, se trata de una medida plenamente justificable.
- Toma colectiva de las decisiones. Aún en un escenario simplificado de interacciones, la toma conjunta de decisiones conduce a resultados globalmente favorables para todas las partes. La propia teoría de los juegos avala este espíritu en situaciones de juegos repetidos un número indefinido de veces incluso cuando los jugadores se comportaran de forma egoísta. En este sentido son también reseñables trabajos como el de Axelrod [8], de los que se deduce que la cooperación es una buena estrategia base mientras que no se detecten violaciones de este comportamiento por parte del resto de los agentes.
- Separación dinámica de los agentes en distintas coaliciones con poca interacción entre sí mediante el uso de herramientas de la teoría de los juegos cooperativos. Si se encuentra que el sistema centralizado puede descomponerse en una serie de subconjuntos con poca interacción entre sí, es posible reducir al mínimo la comunicación entre dichos subconjuntos. Globalmente esto implica un importante ahorro de comunicaciones.

Además del ahorro de comunicaciones, que es el objetivo principal de este trabajo, en la tesis se ha buscado trasponer resultados del problema del control distribuido al problema de estimación distribuido, siempre bajo el prisma de la teoría de juegos.

A.4 Estructura de la presente tesis doctoral

El resto de la tesis está organizado de la siguiente manera:

- **Control predictivo distribuido y la teoría de los juegos**. En este capítulo se estudia en profundidad el estado del arte del control predictivo distribuido, clasificando los diferentes algoritmos propuestos en función del número de comunicaciones empleadas y del tipo de información intercambiada por los agentes.

- **Control predictivo distribuido para dos agentes basado en la teoría de los juegos**. La interacción entre dos agentes se simplifica mediante un sencillo juego en forma estratégica, que reduce el número de acciones de control posibles a tres para cada uno de los agentes. Estas tres opciones consisten en cooperar con el otro agente, comportarse de forma egoísta o acogerse a una alternativa neutral que garantiza la estabilidad del esquema. Los agentes escogen en todo momento la mejor acción desde el punto de vista del coste global.

- **Control predictivo distribuido para N agentes basado en negociación de los agentes**. La explosión combinacional impide generalizar el esquema anterior para un número cualquiera de agentes. Por ello, se desarrolla un esquema simplificado en el que cada agente realiza propuestas al resto. Aquellos agentes afectados por la propuesta responden cuantificando el beneficio o perjucio que la propuesta les causará en caso de ser aceptada. Solo se aceptan aquellas propuestas que disminuyen el coste global del sistema. En caso de que no haya ningún tipo de acuerdo se implementa una acción de control que garantiza la estabilidad del sistema.

- **Filtro de Kalman distribuido**. Dado que el problema de estimación es el problema dual al del control, resulta natural aplicar técnicas de control distribuido a la estimación distribuida. En este capítulo se reduce el problema de estimación de estado a un problema de programación dinámica que es distribuido entre los agentes gracias a la descomposición dual del mismo.

- **Aplicación de la teoría de los juegos cooperativos al control y la estimación de sistemas distribuidos**. La teoría de los juegos cooperativos proporciona herramientas matemáticas muy apropiadas para el análisis de situaciones de conflicto en las que los agentes pueden llegar a algún tipo de acuerdo. Este capítulo de la tesis transpone resultados de esta rama de la teoría de juegos al ámbito del control distribuido. Gracias a ello se desarrolla un esquema de control para la gestión dinámica de los enlaces que componen una red y se proporciona un método para evaluar la importancia relativa de los agentes y enlaces de la red de control.

- **Conclusiones**. La tesis finaliza con un capítulo que analiza las mayores contribuciones de la misma y, adicionalmente, señala líneas de investigación futuras de control distribuido.

A.5 Contribuciones al estado del arte

Una vez ubicada la tesis en la intersección entre los campos de control de sistemas distribuidos y la teoría de los juegos, se repasarán en este apartado las principales aportaciones al estado del arte realizadas y se señalarán también las publicaciones que se han originado a partir del presente trabajo.

Normalmente, cuando se abarca un problema distribuido, lo normal es comenzar por un caso sencillo. Por ese motivo es habitual comenzar estudiando qué sucede cuando se tienen a dos agentes interactuando entre sí. Este fue el primer paso que se dio en la tesis, fruto del cual se proporcionó un esquema de control predictivo distribuido en pocas comunicaciones basado en la teoría de los juegos. Este esquema fue puesto a prueba para sistemas descritos externamente mediante funciones de transferencia [51] y para sistemas expresados en espacio de estados [53, 52]. El esquema de control, además de presentar un rendimiento razonablemente alto para los escasos ciclos de comunicación empleados, garantiza la estabilidad del sistema de control bajo ciertas hiptesis que se verán más adelante en la tesis. Además de los tres artículos citados de congreso, el esquema de control para dos agentes también ha sido publicado en una revista internacional [54] y ha participado en una comparativa de esquemas de control distribuido que se encuentra siendo evaluada en otro artículo de revista [5].

Una vez estudiado el problema para dos agentes, se estudió el caso general de un sistema compuesto por un nmero cualquiera de agentes. Con objeto de evitar la explosión combinacional propia de la interacción de un número indefinido de agentes, se ha desarrollado un algoritmo sencillo de negociación capaz de proporcionar buenas prestaciones de control con un número bajo de comunicaciones. Este trabajo ha sido enviado a [57] y a [58].

Además de estos esquemas originales de control predictivo distribuido, se han realizado otras contribuciones al estado del arte que gozan de un carácter más general, es decir, que pueden ser utilizadas como complemento a otras técnicas de control. En este sentido es especialmente novedosa la aplicación de herramientas de la teoría de los juegos cooperativos para la descomposición dinámica de los agentes de un sistema en diferentes grupos con poca interacción entre sí. Este trabajo ha sido enviado a [56] y [48].

Finalmente, se han realizado también aportaciones al problema dual del control: la estimación. En concreto se ha desarrollado en conjunción con el profesor Anders Rantzer del

LTH una técnica de estimación distribuida basada en la descomposición dual del problema de estimación. Este trabajo ha sido enviado a [50]. Asimismo se ha empleado también la descomposición del sistema en coaliciones para este mismo problema en [48].

Bibliography

[1] T. Alamo, D. Muñoz de la Peña, D. Limon, and E.F. Camacho. Constrained min-max predictive control: Modifications of the objective function leading to polynomial complexity. *IEEE Transactions on Automatic Control*, 50(5):710–714, May 2005.

[2] A. Alessio and A. Bemporad. Decentralized model predictive control of constrained linear systems. In *Proceedings of the 2007 European Control Conference*, pages 2813–2818, 2007.

[3] A. Alessio and A. Bemporad. Stability conditions for decentralized model predictive control under packet drop communication. In *Proceedings of the 2008 American Control Conference*, pages 3577–3582, 2008.

[4] Peter Alriksson. *State Estimation for Distributed and Hybrid Systems*. PhD thesis, Department of Automatic Control, Lund University, September 2008.

[5] I. Alvarado, D. Limon, D. Muñoz de la Peña, J. M. Maestre, F. Valencia, H. Scheu, R. R. Negenborn, M. A. Ridao, B. De Schutter, J. Espinosa, and Marquardt W. A comparative analysis of distributed MPC techniques applied to the hd-mpc four tank benchmark. *Submitted to Journal of Process Control*, 2010.

[6] I. Alvarado Aldea, D. Limón Marruedo, W. M. Garca Gabn, T. Alamo Cantarero, and E. F. Camacho. An educational plant based on the quadruple-tank process. In *Preprints of the 7th IFAC Symposium on Advances in Control Education*, 2006.

[7] S. Andong, W. Yuangang, and Q. Guoqing. Admissible measurement noise of variance-constrained satisfactory filter. In *Proceeding of the 3rd World Congress on Intelligent Control and Automation*, 2000.

[8] Robert Axelrod. *The Evolution of Cooperation*. Penguin Group, 1990.

[9] T. Basar and G. J. Olsder. *Dynamic Noncooperative Game Theory*. Society for Industrial and Applied Mathematics, Philadelphia, 1999.

[10] R.E. Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[11] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, 1997.

[12] E.F. Camacho and C. Bordons. *Model Predictive Control in the Process Industry. Second Edition.* Springer-Verlag, London, England, 2004.

[13] E. Camponogara. *Controlling Networks with Collaborative Nets*. PhD thesis, Carnegie Mellon University, September 2000.

[14] E. Camponogara and B. Barcelos de Oliveira. Distributed optimization for model predictive control of linear-dynamic networks. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 39:1331–1338, 2009.

[15] E. Camponogara, D. Jia, B. H. Krogh, and S. Talukdar. Distributed model predictive control. *IEEE Control Systems Magazine*, 22:44–52, 2002.

[16] V. T. Chow. *Open-Channel Hydraulics*. Mc-Graw-Hill Book, 1859.

[17] J. M. Coron, B. d'Andrea Novel, and G. Bastin. A lyapunov approach to control irrigation canals modeled by saint-venant equations. 1999.

[18] Henry Cox. On the estimation of state variables and parameters for noisy dynamic systems. *IEEE Transactions on automatic control*, 9:5–12, 1964.

[19] L. Dai and K.J. Aström. Dynamic matrix control of a quadruple tank process. In *IFAC World Congress*, 1999.

[20] C. Canudas de Wit. Invited session on advances in networked controlled systems. In *Proceedings of the 25th American Control Conference*, 2006.

[21] P. Deutsch. 8 fallacies of networking, 1992.

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

[23] W. B. Dunbar and S. Desa. Distributed MPC for dynamic supply chain management. In *Int. Workshop on Assessment and Future Directions of NMPC, Freudenstadt-Lauterbad, Germany, 26-30*, 2005.

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

[25] E.P. Gatzke, E.S. Meadows, C. Wang, and F.J. Doyle III. Model based control of a four-tank system. *Computers & Chemical Engineering*, 24:1503–1509, 2000.

[26] E. G. Gilbert and K. T. Tan. Linear systems with state and control constraints: the theory andapplication of maximal output admissible sets. *IEEE Transactions on Automatic Control*, 36:1008–1020, 1991.

[27] L. Giovanini and J. Balderud. Game approach to distributed model predictive control. In *Proceedings of the 2006 International Control Conference*, 2006.

[28] P. Giselsson and A. Rantzer. Distributed model predictive control with suboptimality and stability guarantees. In *Proceedings of the 2010 Control and Decision Conference*, 2010.

[29] J. C. Harsanyi. A simplified bargaining model for the n-person cooperative game. *International Economic Review*, 4(2):194–220, 1963.

[30] Eric L. Haseltine and James B. Rawlings. Critical evaluation of extended kalman filter and moving horizon estimation. *Industrial and Engineering Chemistry Research*, 44:2451–2460, 2005.

[31] Anne-Kathrin Hess. A distributed kalman filter algorithm for self-localization of mobile devices. Technical report, Department of Automatic Control, Lund University, Sweden, 2009.

[32] M.B. Jamoom, E. Feron, and M.W. McConley. Optimal distributed actuator control grouping schemes. In *Proceedings of the 37th IEEE Conference on Decision and Control*, volume 2, pages 1900–1905, 1998.

[33] D. Jia and B. Krogh. Distributed model predictive control. In *Proceedings of the 2001 American Control Conference*, pages 4507–4512, 2001.

[34] D. Jia and B. Krogh. Min-max feedback model predictive control for distributed control with communication. In *Proceedings of the American Control Conference*, pages 4507–4512, Anchorage, 2002.

[35] K. H. Johansson. The quadruple-tank process. *IEEE Transactions on Control Systems Technology*, 8:456–465, 2000.

[36] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME-Journal of Basic Engineering*, 82: Series D:35–45, 1960.

[37] E. C. Kerrigan. *Robust Constraint Satisfaction: Invariant Sets and Predictive Control*. PhD thesis, Department of Engineering, University of Cambridge, UK, November 2000.

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

[39] I. Kolmanovsky and E. G. Gilbert. Theory and computation of disturbance invariant sets for discrete-time linear invariant systems. *Mathematical Problems in Engineering: Theory, Methods and Applications*, 4:317–367, 1998.

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

[41] M. Lazar, W. P. M. H. Heemels, D. Muñoz de la Peña, and T. Alamo. Further results on robust MPC using linear matrix inequalities. In *Proceedings of Int. Workshop on Assessment and Future Directions of NMPC*, Pavia, Italy, September 2008.

[42] M. Lazar, W. P. M. H. Heemels, S. Weiland, and A. Bemporad. Stabilizing model predictive control of hybrid systems. *IEEE Transactions on Automatic Control*, 51:1813–1818, 2006.

[43] Shaoyuan Li, Yan Zhang, and Quanmin Zhu. Nash-optimization enhanced distributed model predictive control applied to the shell benchmark problem. *Inf. Sci. Inf. Comput. Sci.*, 170(2-4):329–349, 2005.

[44] D. Limon, I. Alvarado, T. Alamo, and E. F. Camacho. MPC for tracking of piece-wise constant references for constrained linear systems. *Automatica*, 44(9):2382–2387, 2008.

[45] J. Liu, D. Muñoz de la Peña, B. Ohran, P. D. Christofides, and J. F. Davis. A two-tier architecture for networked process control. *Chemical Engineering Science*, 63:5349–5409, 2008.

[46] J. Liu, D. Mu noz de la Peña, and P. D. Christofides. Distributed model predictive control of nonlinear process systems. *AIChE J.*, 55:1171–1184, 2009.

[47] A. Van den Nouweland M. Slikker. *Social and Economics Networks in Cooperative Game Theory*. Kluwer Academic Publishers, 2001.

[48] J. M. Maestre. Applications of cooperative games to distributed control and estimation. In *Junior FeedNetBack Workshop, Annecy.*, 2010.

[49] J. M. Maestre and E. F. Camacho. Smart home interoperability: the domoesi project approach. *International Journal of Smart Home*, 3:31–44, 2009.

[50] J. M. Maestre, P. Giselsson, and A. Rantzer. Distributed receding horizon kalman filter. In *Proceedings of the 2010 Control and Decision Conference*, 2010.

[51] J. M. Maestre, D. Muñoz de la Peña, and E. F. Camacho. Distributed model predictive control in few communication cycles. In *Proceedings of the American Control Conference*, pages 2797–2802, 2009.

[52] J. M. Maestre, D. Muñoz de la Peña, and E. F. Camacho. Distributed MPC: a supply chain case study. In *Proceedings of Conference on Decision and Control. Accepted for publication*, 2009.

[53] J. M. Maestre, D. Muñoz de la Peña, and E. F. Camacho. Distributed MPC based on a cooperative game. In *Proceedings of Conference of Decision and Control. Accepted for publication*, 2009.

[54] J. M. Maestre, David Muñoz de la Peña, and E. F. Camacho. Distributed model predictive control based on a cooperative game. *Optimal Control Applications and Methods*, page In press, 2010.

[55] J.M. Maestre, D. Muñoz de la Peña, and E. F. Camacho. An application of cooperative game theory to distributed control. In *Submitted to the 18th IFAC World Congress*, 2010.

[56] J.M. Maestre, D. Muñoz de la Peña, and E. F. Camacho. An application of cooperative game theory to distributed control. *Submitted to Transactions of Automatic Control*, 2010.

[57] J.M. Maestre, D. Muñoz de la Peña, and E. F. Camacho. Distributed model predictive control based on agent negotiation. *Submitted to Journal of Process Control*, 2010.

[58] J.M. Maestre, D. Muñoz de la Peña, and E. F. Camacho. Distributed model predictive control based on agent negotiation. In *Submitted to 2011 American Control Conference*, 2011.

[59] L. Magni, G. De Nicolao, R. Scattolini, and F. Allgower. Robust MPC for nonlinear discrete-time systems. *International Journal of Robust and Nonlinear Control*, 13:229–246, 2003.

[60] L. Magni and R. Scattolini. Stabilizing decentralized model predictive control of nonlinear systems. *Automatica*, 42:1231–1236, 2006.

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

[62] M. Mercangoz and F. J. Doyle. Distributed model predictive control of an experimental four-tank system. *Journal of Process Control*, 17:297–308, 2007.

[63] P. Millan, L. Orihuela, C. Vivas, and F.R. Rubio. Improved delay-dependent stability criterion for uncertain networked control systems with induced time-varying delays. In *FeedNetback Workshop*, 2009.

[64] F. J. Muros Ponce, J. M. Maestre Torreblanca, and E. F. Camacho. Estudio de robustez frente a retardos y pérdida de datos de una estrategia DMPC basada en pocos ciclos de comunicación. In *Actas de las XXIX Jornadas de Automática*, 2008.

[65] R. B. Myerson. Graphs and cooperation in games. *Math. Oper. Res.*, 2:225–229, 1977.

[66] R. B. Myerson. *Game Theory. Analysis of Conflict.* Hardvard University Press, 1997.

[67] N. Nader Motee and B. Sayyar-Rodsari. Optimal partitioning in distributed model predictive control. In *Proceedings of the 2003 American Control Conference*, volume 6, pages 5300–5305, 2003.

[68] J. F. Nash. Equilibrium points in n-person games. In *Proceedings of the National Academy of Science of United States of America*, volume 36, pages 48–49, june 1950.

[69] J. F. Nash. Two-person cooperative games. *Econometrica*, 21(1):128–140, 1953.

[70] R. R. Negenborn, P. J. Van Overloop, T. Keviczky, and B. De Schutter. Distributed model predictive control of irrigation canals. *Networks and Heterogeneus Media*, 4:359–380, 2009.

[71] R.R. Negenborn. *Multi-Agent Model Predictive Control with Applications to Power Networks*. PhD thesis, Delft University of Technology, 2007.

[72] R.R. Negenborn, B. De Schutter, and J. Hellendoorn. Multi-agent model predictive control for transportation networks: Serial versus parallel schemes. *Engineering Applications of Artificial Intelligence*, 21(3):353–366, April 2008.

[73] P. Neumann. Communication in industrial automation - what is going on? *Control Engineering Practice*, 15:1332–1347, 2007.

[74] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer-Verlag, 1999.

[75] Owen G. P. Borm and S. Tijs. On the position value for communication situations. *SIAM Journal on Discrete Mathematics*, 5:305–320, 1992.

[76] H. J. M. Peters. *Axiomatic bargaining game theory*. Kluwer Academic Publishers, 1992.

[77] W. Poundstone. *Prisoner's Dilemma: John Von Neumann, Game Theory, and the Puzzle of the Bomb*. Anchor Books, 1992.

[78] B. De Schutter R. R. Negenborn and H. Hellendoorn. Multi-agent model predictive control of transportation networks. In *Proceedings of the 2006 IEEE International Conference on Networking, Sensing and Control (ICNSC 2006)*, pages pp. 296–301, Ft. Lauderdale, Florida, April 2006.

[79] D. M. Raimondo, L. Magni, and R. Scattolini. Decentralized MPC of nonlinear system: An input-to-state stability approach. *Int. J. Robust Nonlinear Control*, 17:1651–1667, 2007.

[80] S. Rakovic, E. Kerrigan, K. Kouramas, and D. Mayne. Robust MPC for nonlinear discrete-time systems. *IEEE Transactions on Automatic Control*, 50:406–410, 2005.

[81] S. V. Rakovic, E. De Santis, and P. Caravani. Invariant equilibria of polytopic games via optimized robust control invariance. In *Proceedings of the 44th IEEE Conference on Decision and Control and the European Control Conference*, pages 7686–7691, Seville, Spain, December 2005.

[82] A. Rantzer. Dynamic dual decomposition for distributed control. In *American Control Conference 2009*, 2009.

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

[84] James B. Rawlings and Brett T. Stewart. Coordinating multiple optimization-based controllers: New opportunities and challenges. *Journal of Process Control*, 18(9):839 – 845, 2008.

[85] J.B. Rawlings and B.R. Bakshi. Particle filtering and moving horizon estimation. *Computers and Chemical Engineering*, 30:1529–1541, 2006.

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

[87] S.I. Roumeliotis and G.A. Bekey. Distributed multirobot localization. *IEEE Transactions on Robotics and Automation*, 18(5):781–795, 2002.

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

[89] H. Scheu, J. Busch, and W. Marquardt. Nonlinear distributed dynamic optimization based on first order sensitivities. In *Proceedings of the American Control Conference*, Baltimore, Maryland, USA, July 2010.

[90] P. O. M. Scokaert, D. Q. Mayne, and J. B. Rawlings. Suboptimal model predictive control (feasibility implies stability). *IEEE Transactions on Automatic Control*, 44:648–654, 1999.

[91] N. Z. Shor. *Minimization Methods for Nondifferentiable functions*. Springer, 1985.

[92] J. D. Sterman. *Business Dynamics - Systems Thinking and Modelling in a Complex World*. McGraw-Hill, New York, 2000.

[93] B. T. Stewart, Venkat A. N., J. B. Rawlings, Wright S. J., and G. Pannocchia. Cooperative distributed model predictive control. *Systems and Control Letters*, 2010.

[94] P. Trodden and A. Richards. Robust distributed model predictive control using tubes. In *Proceedings of 2006 American Control Conference*, June 2006.

[95] R. Vadigepalli, E.P. Gatzke, E.S. Meadows, C. Wang, and F.J. Doyle III. Robust control of a multivariable experimental 4-tank system. *Computers & Chemical Engineering*, 24:1503–1509, 2000.

[96] H. R. Varian. Economic mechanism design for computerized agents. In *Proceedings of the First USENIX Workshop on Electronic Commerce*, 1995.

[97] H. R. Varian. *Intermediate Microeconomics: A Modern Approach.* W.W. Norton & Co., 2005.

[98] A. N. Venkat, J. B. Rawlings, and S. J. Wright. Stability and optimality of distributed model predictive control. In *Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference ECC 2005*, pages 6680–6685, Seville, Spain, 2005.

[99] Aswin N. Venkat. *Distributed Model Predictive Control: Theory and Applications*. PhD thesis, University of Wisconsin-Madison, 2006.

[100] J. Von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1944.

[101] T. C. Yang. Networked control systems: a brief survey. In *IEE Procedings- Control Theory and Applications*, volume 152, pages 403–412, 2006.

[102] A. Zafra-Cabeza, J. M. Maestre, M. A. Ridao, E. F. Camacho, and L. Sánchez. A hierarchical distributed model predictive control approach in irrigation canals: A risk mitigation perspective. *Submitted to Journal of Process Control*, 2010.

[103] A. Zafra-Cabeza, J. M. Maestre, M. A. Ridao, E. F. Camacho, and L. Sánchez. Hierarchical distributed model predictive control for risk mitigation: An irrigation canal case study. In *Submitted to 2011 American Control Conference*, 2011.