Improvement of the sensory stimuli restitution on driving simulator

Martin Soyer

université PARIS-SACLAY

Improvement of the sensory stimuli restitution on driving simulator

Amélioration de la restitution des stimuli sensoriels sur simulateur de conduite

Thèse de doctorat de l'université Paris-Saclay

École doctorale n° 580, Sciences et Technologies de l'Information et de la Communication (STIC)
Spécialité de doctorat: Automatique
Graduate School : Sciences de l'ingénieur et des systèmes (SIS),
Référent : CentraleSupélec

Thèse préparée dans l'unité de recherche Université Paris-Saclay, CNRS, CentraleSupélec, Laboratoire des signaux et systèmes, 91190, Gif-sur-Yvette, France, sous la direction de Sorin OLARU,professeur et le co-encadrement de Zhou FANG, Immersive driving simulation expert

Thèse soutenue à Paris-Saclay, le 20 juillet 2022, par

Martin SOYER

Composition du jury

Sihem TEBBANI Professeure, CentraleSupélec - Université Paris-Saclay	Présidente
Nikolaos ATHANASOPOULOS Senior Professor, Queen's University Belfast	Rapporteur & Examinateur
Guillaume COLIN Professeur, Université d'Orléans	Rapporteur & Examinateur
Morten HOVD Professeur, Norwegian University of Science and Technology	Examinateur
Antonio FERAMOSCA Associate Professor, University of Bergamo	Examinateur
Ewa GIREJKO Assistant Professor, Bialystok University of Technology	Examinatrice
Sorin OLARU Professeur, CentraleSupélec - Université Paris-Saclay	Directeur de thèse
Zhou FANG Immersive driving simulation expert, Renault	Encadrant

THESE DE DOCTORAT
NNT : 2022UPAST103

I

«Echangeriez-vous votre vie réelle, marquée par des frustrations et des déceptions, des succès partiels et des rêves inaccomplis, contre une vie d'expériences désirables mais complètement artificielles, provoquées par des moyens chimiques ou mécaniques ?»

Ruwen Ogien, *L'influence de l'odeur des croissants chauds sur la bonté humaine*

Acknowledgment/Remerciements

This thesis work results from the direct or indirect collaboration of many people that I want to thank here.

I first thank Pr. Sorin Olaru for guiding me during this thesis, particularly for sharing with me the taste of research, teaching and demanding work. These three years were extremely stimulating thanks to him.
I want to thank the department of driving simulation of Renault for welcoming me among them. In particular I would like to thank Zhou Fang for introducing me this exciting topic and this stimulating environment.

I would like to express my gratitude to all the members of the jury, to Pr. Nikolaos Athanasopoulos and Pr. Guillaume Colin for accepting to review the present manuscript, to Pr. Morten Hovd, Pr. Antonio Feramosca, Pr. Ewa Girejko and Pr. Carlos Trabuco Dorea for accepting to be part of the committee and finally to Pr. Sihem Tebbani for presiding the jury. I thank them for graduating me, I will do everything I can to be worthy of this PhD.

Ces 3 années au sein de l'équipe SYCOMORE du Laboratoire des Signaux et Systèmes de CentraleSupélec ont été stimulantes et agréables. Je remercie l'équipe enseignante ainsi que les post-doctorants et les actuels doctorants que j'ai pu croiser au fil de cette thèse. Je remercie chaleureusement les énergumènes qui m'ont accompagnés dans cette aventure: Maxime, Geoffrey, Jeremy, Thomas, Vincent, Dario, Mathieu et Baptiste.

Mon choix de commencer ce travail de thèse est né d'une déconvenue professionnelle et à des doutes, ainsi je tiens à remercier mes parents pour leur soutien inconditionnel et pour m'avoir soutenu dans ce choix de carrière que je m'aveuglais à ne pas faire. Je remercie également ma magicienne de soeur ainsi que ses deux petits monstres.

Je veux remercier mes amis Alban, Arthur, Laure, Marie-Adélaïde, Marion, Nicolas et Thibault qui m'ont accompagné tout au long de l'aventure "Supélec" (et même quasiment littéralement jusqu'à la veille de la soutenance !). Je remercie également Caroline ainsi que toutes les personnes que j'aurai pu oublier de citer ici m'ayant accompagné ou croisé durant ce travail et qui, j'espère, se reconnaîtront à la lecture de ces lignes.

Enfin je remercie spécialement Joy pour son soutien et son amour.

Contents

Chapter 1

Introduction

Contents

1.1 Driving Simulation in the automotive industry

This section will provide a global description of the framework of this thesis, the driving simulation through its role in the automotive and research as well as its history from first flight simulators to the most recent high performance driving simulations and their realization within industrial integrated platforms. We additionally explicit the electromechanical composition of the simulators studied during this thesis.

1.1.1 What is Driving Simulation ?

Global framework

The advances in automotive technologies can be measured with different features such as:

- driving confort
- safety (number of accidents, wounds, deaths,...)
- pollution (CO_2 emissions, nitrogen oxides, airbornes)
- price-quality ratio

Thus, when a car manufacturer develops a new vehicle or a driving assistance system (in the recent trends of developments), it has to guarantee the norms satisfactions, the clients expectations as well as industrial and commercial requirements. However, the number, the complexity and the hazardousness of the potential driving situations represent a serious obstacles for the validation of new prototypes.

The simulation consists in reproducing artificially different driving scenarios in a parametrizable framework, using a restricted workspace (sometimes completely virtual) all by guaranteeing a high guarantee of safety. From this definition, it becomes obvious that the simulation at the level of a prototype or a particular component becomes a valuable part of the design chain.

The driver was and it will remain the center of attention until the complete automatization of the driving task, the internal system of perception of information from environment (stimuli) and the decision-making process are the core of the driving. A **simulated experience** consequently has to reproduce the driver's sensations.

We propose the following definition for the driving simulation:

Definition 1.1: Driving Simulation

The **driving simulation** is the collection of physical and numeric processes allowing the reproduction of driver's sensations in a safe and constrained environment.

The key concepts pertaining to the above are developed next.

Sensation restitution

Humans receive information of their environment through senses *i.e* a system of specific cells and organs which convert stimuli towards electric signals that are processed and interpreted by the nervous system. The different senses known to this

day are: sight, hearing, smell, touch, taste, thermoception (sense of temperature), nociception (sense of pain), proprioception (sense of body positioning), equilibrioception (sense of equilibrium and spatial orientation). The five first senses cited above corresponds to a widespread idea of senses that had been given by Aristotle during the IVth century BCE [Aristotle, BCE], the other ones are commonly accepted as senses in physiology and are important to consider. It is also clear that some senses are more solicited while driving, particularly the sight. Indeed, the driver focuses on and reacts to visual stimuli such as signs, pedestrian crossings or other drivers behaviors, but auditive stimulation is also important in some situations such as klaxons or sirens. However, the driver also feels the accelerations through the equilibrioception sensed by the vestibular system in the inner ear (see Section 2.4.1) and this aspect gains in interest whenever comfort and passengers conditions are considered in the automotive design. More precisely, longitudinal accelerations are felt when pushing the acceleration pedal while lateral ones are felt when turning thanks to the centrifugal force.

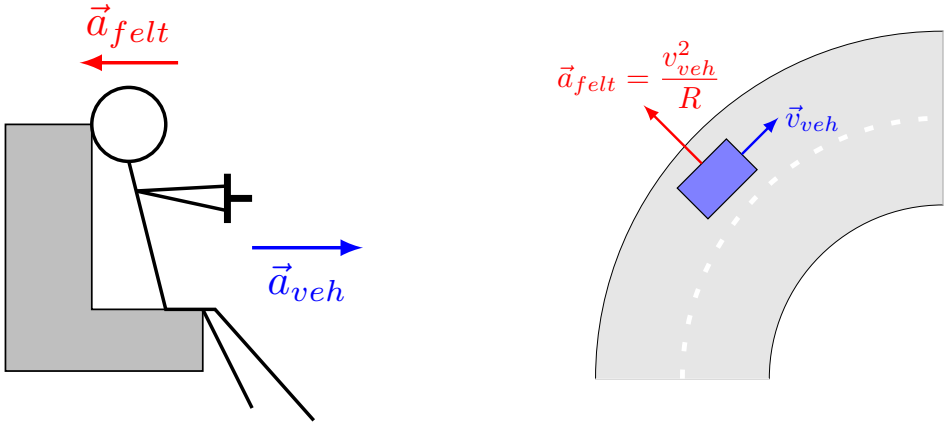

Figure 1.1: Acceleration Felt linearly (left) and during a turn (right)

Safe and restricted workspace

Driving always presents safety risks due to inadvertency, weather or limited reaction to high speeds for example, consequently testing prototypes on a road or on a highway may increase the risks and the financial costs. So it is interesting to virtually simulate the vehicle or functionalities to study their viability and fiability. A simulator drived by a human (also identified by the term "Driver In the Loop" or "DIL") should stimulate senses cited previously. To this purpose, the sight can

be stimulated thanks to a screen or a virtual reality helmet displaying the road, the traffic and the differents events occuring during a scenario with a real time update according to the driver's actions. The sound restitution can be done with a classic sound system (speakers or audio headset).

The restitution of accelerations is the third important aspect for such a simulator and will represent the main topic of this thesis. Since the acceleration describes variations in movement, the reproduction of an acceleration needs a workspace of a specific length. If we consider a scenario of accelerating from 0 to 100km/h in $\Delta t = 10$s with a constant acceleration a_{veh} on a highway for example, the distance to be traveled can be estimate by integration:

$$L = \frac{1}{2} a_{veh} \Delta t^2 \approx 140m \tag{1.1}$$

This first analysis points to the fact that the considered scenario may imply large distances of manoeuver, spaning tens, hundred or thousands of meters whenever the simulator needs to be mechanically displaced. A solution used to economize this distance is to exploit the gravitational field by tilting the driver and the viewing system, thus a component of the gravitational field is perceived by the driver as a linear acceleration while there is no displacement. This technique is called **Tilt Coordination** in the literature and is illustrated on Figure.1.2 where the projection of the gravitational force is shown to be sensed even if the driver is in a static configuration. The simulators allowing acceleration restitution are called "dynamic simulators" in opposition with the "static simulators".

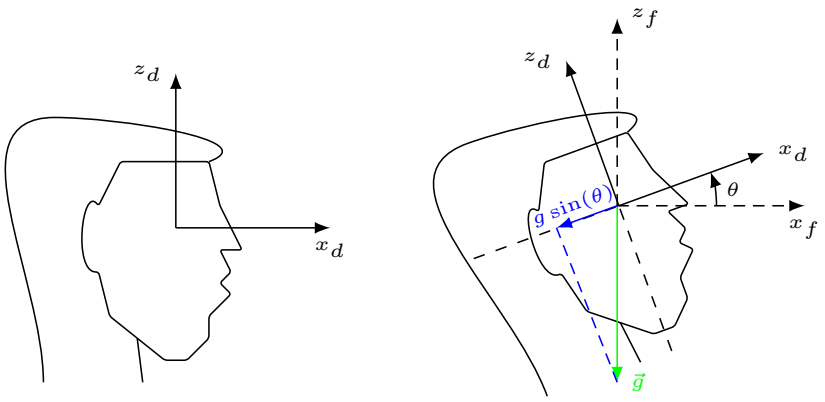

Figure 1.2: Illustration of the tilt coordination techniques

1.1.2 Driving simulators

Static simulators

Driving simulators don't have the same purpose and the same cost. Since the visual rendering is a high priority and has to be available in any configuration, the investment is often concentrated on this chapter and most simulators don't restitute the accelerations. Those simulators are said to be static and present a better transportability potential and are less costly as long as the development is mainly related to graphical processing algorithms. However, as it can be easily imagined, driving simulations that intend to replicate the stimuli on the driver and the related safety and manoeuvrability aspects need to include a dynamic aspect.

Dynamic simulator

Dynamic simulators are equipped with a system allowing the movement of the driver as a whole. In this work, we focus on high performance driving simulators whose composition is given on Figure.1.3 and includes:

- The screen where the virtual environment is displayed
- The cabin which is a real car 's one (A Renault Twingo on ULTIMATE)
- The hexapod (also called Stewart platform or Cough-Stewart patform) tilts the cabin and the screen to operate the tilt coordination technique
- The rails that move the platform laterally and longitudinally in order to generate rapid movements

In this work, for a automatic control consideration, we will identify rails and hexapod as actuators. The operation of a dynamic driving simulator is summarized in Figure.1.4 and its functioning is resumed step by step as follows:

1. The driver in the cabin reacts to visual stimuli displayed on screen by pushing the acceleration or braking pedal or by turning the steering wheel.

2. The generated signals are collected by means of actuators located on the plateform/driver and processed by the driving simulation software. This latter block computes the accelerations (lateral and longitudinal) that should be felt by the driver thanks to a model of the vehicle dynamics.

3. The acceleration signals are used as reference to be followed by the real-time control architecture of the platform (which will be explicited in Section 2.1) actuating elements for the rail and hexapod movements with the aim to make the driver feeling the expected sensations.

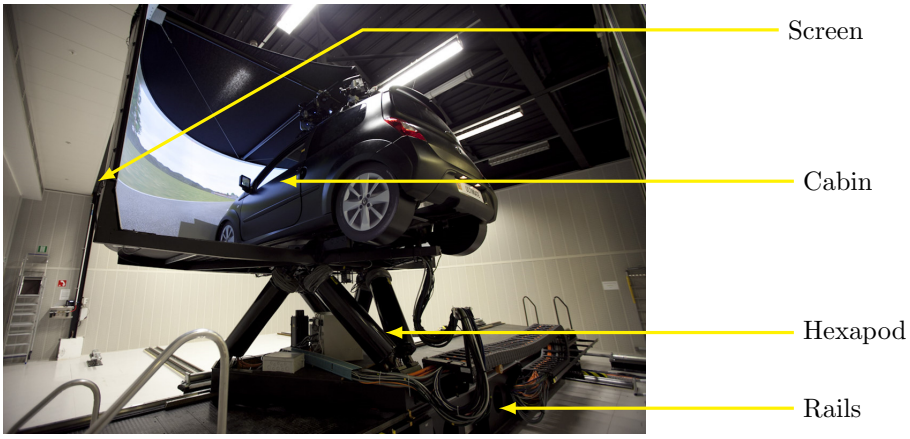

Figure 1.3: Main components of a dynamic driving simulator (in the present case, the Renault's ULTIMATE)

1.1.3 Practical role of Driving Simulation

The investments on driving simulators allows car manufacturers to support the development and design of new functionalities on their vehicles safely and for a less cost. In parallel, some research institutes have an interest in driving simulation with the aim to study the drivers behaviors. We explicit here some roles of the driving simulation.

Advanced Driving Assistance Systems (ADAS)

ADAS systems are electronic devices implemented in vehicles with the aim to prevent the driver from having accidents. They can also allow the driver to focus on what happens on the road by sensing hazardousness events (weather, pedestrian crossing, overtaking, distance from other vehicles,...). Those systems are more and more implemented in new car models, here are some examples:

- Emergency brake assists
- Adaptative cruise control
- Pedestrian detection
- Automatic parking

Autonomous driving

One of the main issues for car manufacturers nowadays is the development of the autonomous car. They aim to develop and sell a car where the user is free of

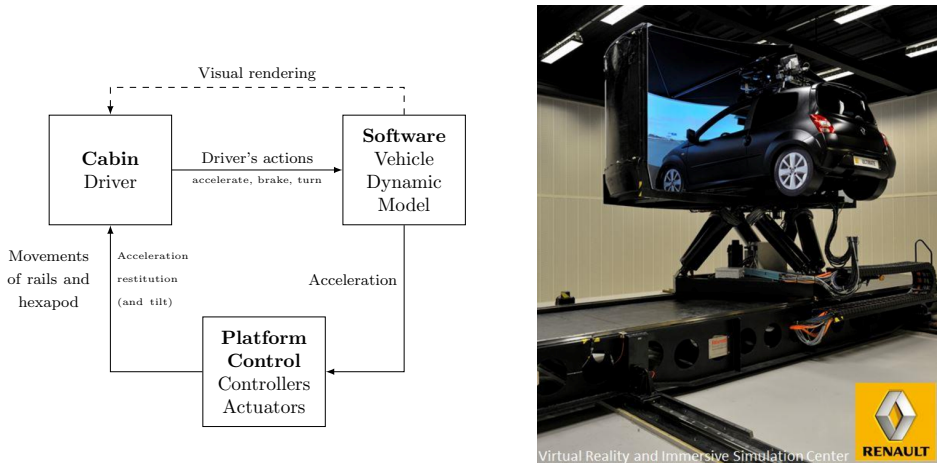

Figure 1.4: Summary of Driving simulation operating (left). Example of a dynamic driving simulator: Renault's ULTIMATE

focusing on driving tasks. Nowadays, there exists some partially autonomous car *i.e* the driver needs to focus on the road ("eyes-on") and have hands on the steering wheel ("hands-on"). An autonomous car can be seen as a vehicle full of ADAS systems which have to be validates step by step by simulation and experimental trials

Research

Simulators can be used to study the driver's behavior in specific situations that are difficult to reproduce on real road. For example [Sato et al., 2019] compare the driver's behavior after a transition automated/manual driving both on simulator and on real road. Thus simulators may give a first idea of the behaviors of a driver when they are subject to unusual events in an (semi-)autonomous car. Recording such human reactions is invaluable given the impossibility of playing such scenarios in real-life tests. In a physiological framework, simulators can provides models of senses fusion in the decision making process. The research field around driving simulation is active with dedicated international conferences such as Driving Simulation & VR Conference where simulation is used in many topics such as ADAS development or perception and human factors understanding.

The structure of high performance driving simulators results from technological advances in transport domain all over the XX^{th} century. We show in the next section that driving simulation inherits from flight simulation and virtual reality.

1.2 History of Driving Simulation

1.2.1 Flight simulators

The history of aviation starts at the beginning of the XXth century with the first flights of the Wright brothers. The quick evolution of this industry in America and in Europe had implied the need to train pilots to get acclimated to the new flight sensation, particularly during the World Wars. In 1910, in France a school of pilots called École Antoinette build a simulator using barrels to simulate lateral movements (see Figure.1.5) where pitch, roll and yaw are actuated manually. In parallel, in the United States, the Breese aircraft Corporation developed an aircraft known as "penguin" with reduced wingspan such that it cannot really take off as explained in [Page, 2000] but allows to familiarize with the flight accelerations and maneuvers.

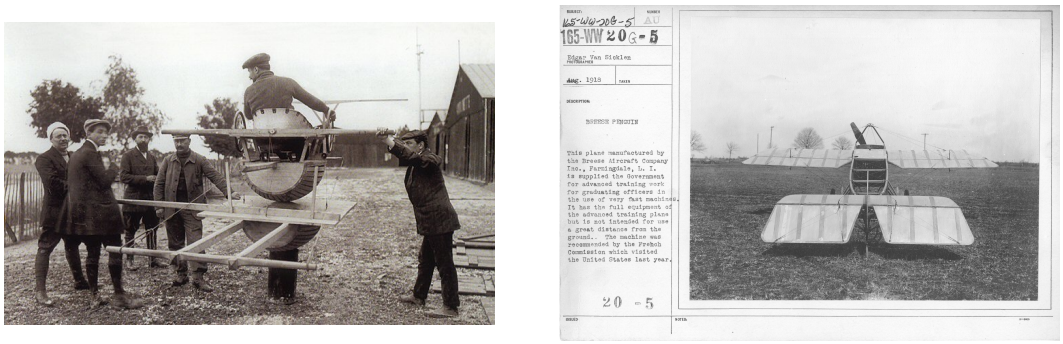

Figure 1.5: Left: "Antoinette" flight simulator (1910). Right: Breese "Penguin" simulator (1918).

In 1931, the entrepreneur Edwin Link (1904-1981) published a patent claiming the invention of a new kind of flight simulator: the Trainer Link (Figure.1.2.1). The simulator has electro-pneumatic actuators and an instrument panel. The trainer Link had been used and enhanced for night navigation training and bombing in preparation of the WWII. By these means it is claimed that during the war, the US Army might had accelerated the training of half a million pilots [De Angelo, 2000]. It has been used until the 1960's when industries began to focus on impacts of aircrafts movements on pilots reactions.

During the 1960's, the NASA developped their own 6 degrees of freedom simulator named AMES Motion generator and used to developed the first control engineering algorithms [Schmidt and Conrad, 1970] (Figure 1.7). In parallel, in 1965, Stewart proposes a dynamic platform for helicopter simulation [Stewart, 1965]. This platform (which is known under his name) is a hexapod actuated by

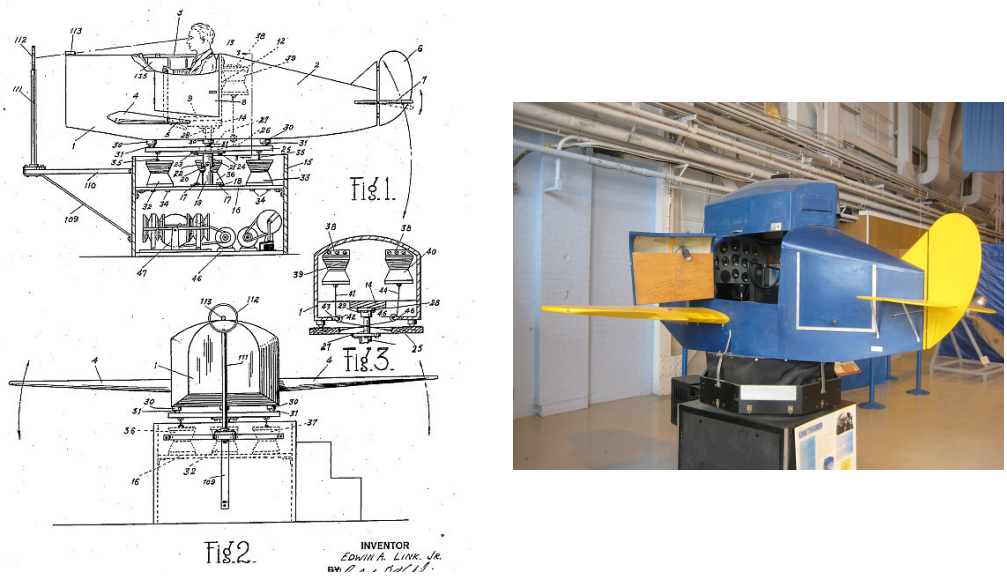

Figure 1.6: Left: Patent of the Trainer Link (1931). Right: Photo of a Trainer Link

electrical pistons. Nowadays this architecture is widely used in flight and driving simulation (Figure 1.7).

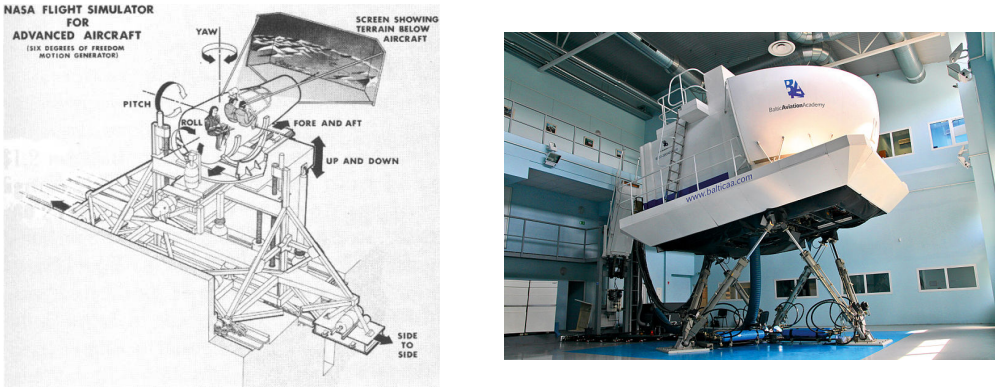

Figure 1.7: Left: NASA AMES motion generator. Right:Baltic Aviation Full Flight simulator

1.2.2 Development of driving simulation

While flight simulators had evoluted in the purpose of training and learning due to the high costs and high risk of flight, the development of driving simulators had focused on the driver's behaviors towards traffic events. Thus, industries and

research centers had prioritized visual stimulation. Indeed, while flight needs a specific focus on cruise instruments (altimeters, pressure and speed sensors,...), driving needs a specific attention on external events (pedestrian crossing, signs, other driver behavior,...). The development of driving simulators is consequently joint with virtual reality one's, in this context one can consider that the first driving simulator is the Morton Heiling's Sensorama whose patent has been granted in 1962 (Figure 1.8). The system projected a movie of a motorcycle travel in a city, visual, auditive and olfactive stimuli were activated according to the script of a scenario [Craig et al., 2009]. During the 1970's, the Virginia Polytechnic Institute and State University (VPI-SU) devised an interactive simulator (Figure 1.8) with fundings of General Motors. The simulator provided a television-based display and a dynamic platform moving laterally, in roll and in yaw [Repa and Wierwille, 1976].

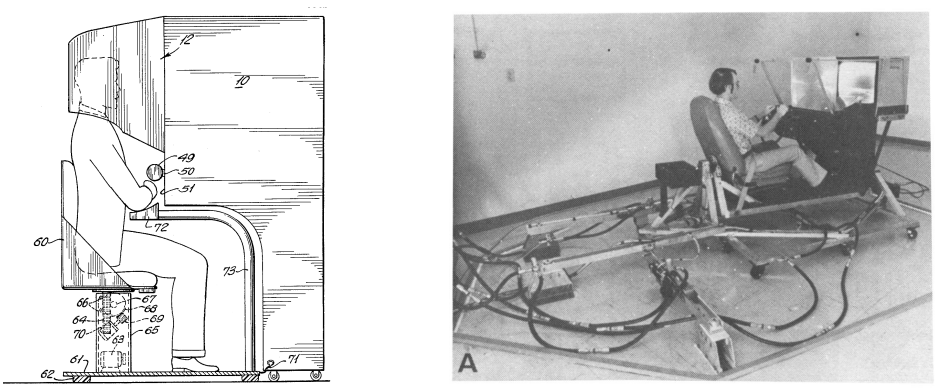

Figure 1.8: Left: Patent of the Sensorama. Right: The VPI-SU simulator

From this decade, the analogic technologies used had been progressively replaced by computers, this evolution allowed in particular a better visual rendering [Fisher et al., 2011]. Thus, in 1983, the US department of Highway Administration through its Human Factors Laboratory began the exploitation of its own simulator Hysim (Figure 1.9). In 1984, the swedish research institute on road safety VTI exploited its dynamic driving simulator [Nordmark, 1984] (Figure 1.10).

In 1985, Daimler-Benz exploits a simulator (Figure 1.11)based on Stewart platform at Berlin. The hexapod is mounted on a lateral rail [Drosdol and Panik, 1985], becoming the most performing driving simulator in the automotive industry and becoming the precursor of the new generation of modern simulators based on rails and hexapods.

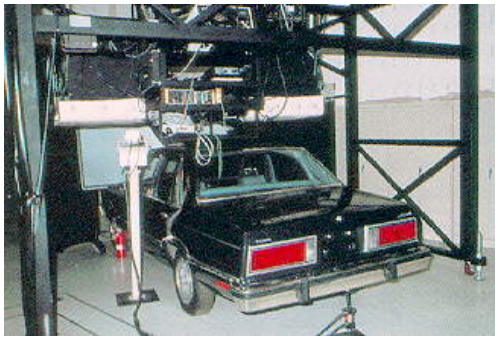

Figure 1.9: Hysim

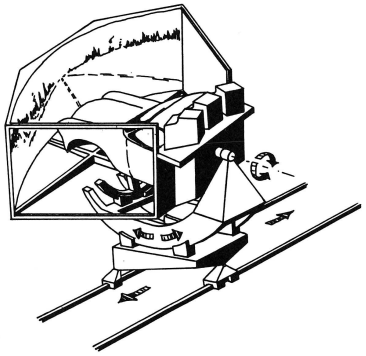

Figure 1.10: The VTI Simulator

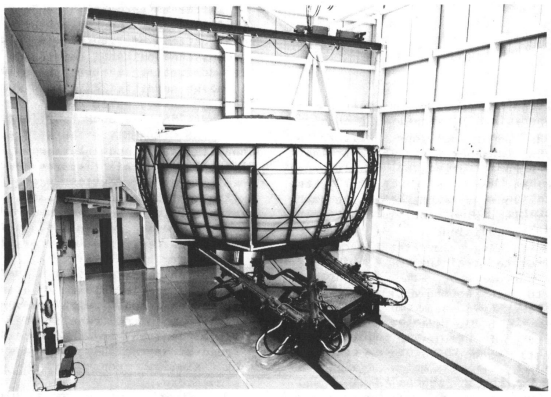

Figure 1.11: Daimler-Benz Simulator

1.2.3 Modern simulators and industrial spread

Since the 1990's, the automotive manufacturers and research institutes have invested in driving simulation, most of them have built and operated their own high performance simulators such as:

- NADS (National Advanced Driving Simulator,2001) of National Highway Traffic Safety Administration of the United States build in the University of Iowa.
- Ford VIRTTEX (VIRtual Test Track EXperiments, 2001)
- Nissan NIDS
- Renault ULTIMATE, 2004
- Toyota, 2007

- Renault ROADS, 2022

Those simulators have benefited from the computational capacity increase of processors and graphic cards for the visual rendering. This engineering field are in the forefront of the developments for the driving simulators but the recent developments show that the interplay with the physiology and the control of the robotic platforms is the key factor for game change in terms of rendering.

Finally, coming closer to the core of the present PhD work, those complex robotic structures need control strategies to achieve their purpose of restituting acceleration feelings, and these strategies have evoluted during the past decades with control theory, computer science and mathematics progress (in terms of modelling). The present thesis aims to contribute to this aspect.

In the following we focus on high performance driving simulators and particularly on the control architecture allowing the driving of rails and hexapod in order to restitute the acceleration sensations expected by the driver. We introduce first the mathematical and control theoretic tools which permit us to address the issues raised in this work.

1.3 Open problems and contribution of the thesis

1.3.1 Open problems

Nowadays, driving simulation is an active field of research with its own conferences and industrial exhibitions addressing trends and open problems in driving immersion by integrating modern features such as autonomous driving. The Figure 1.12 illustrates the different components of the driving simulation field and how they are linked. The software plans displays the visual rendering using road (curve,length, ...) and navigation (drivers actions, braking, steering wheel turn, ...) data.

The visual stimulation is a major part of the simulation and can be compared to a video game design, practically the development of the visual rendering seems to follow this industry by developing immersion through head-mounted display such as virtual-reality helmets [Parduzi et al., 2020, Zöller et al., 2019, Ivleva et al., 2019, Hartfiel and Stark, 2019].

The software also computes the acceleration profiles that have to be rendered by the platform by using a model of the vehicle dynamics. Then those acceleration profiles are processed in the way to drive the simulator.

The present thesis mainly focuses on this last part and will be detailed and developed from the Chapter 2, however one can notice the major interest of the

community for this part in the past years [Fang et al., 2019, Ronellenfitsch et al., 2019, Ellensohn et al., 2019, Kolff et al., 2020]. The platform driving is controlled thanks to a feedback which represents the driver perception. The previous features are designable and tunable from an engineering point of view. From this point, we can distinguish two cases:

- **Autonomous driving**: the driver does not interact with the simulation
- **Human driving**: the driver interacts with the simulator and the simulation in reaction to the visual rendering. In this case, the quality of the simulation is directly linked to the model reliability.

In both cases, the mismatch between perception model and real sensation as well as the visual desynchronization between the driver expectations and the actual sensation may have undesirable consequences on the user comfort causing uneasiness commonly named *Motion Sickness* which is still a research topic [Hogerbrug et al., 2020, Reuten et al., 2020].

Thus, the field of driving simulation is wide and follows many vectors of development such as the growth of graphical performance of computers or the better comprehension of the driver behavior from a neuroscience point of view. The automotive industry has to face major issues with electrification and automatization which demand a capacity of projection and experiments upstream of the production, this horizon can be provided by the simulation. This thesis emerged in this framework and was driven by Renault's will to explore the fundamental limitations of driving simulations and to develop the methodological tools that optimize the real-time control of the high performance driving simulators.

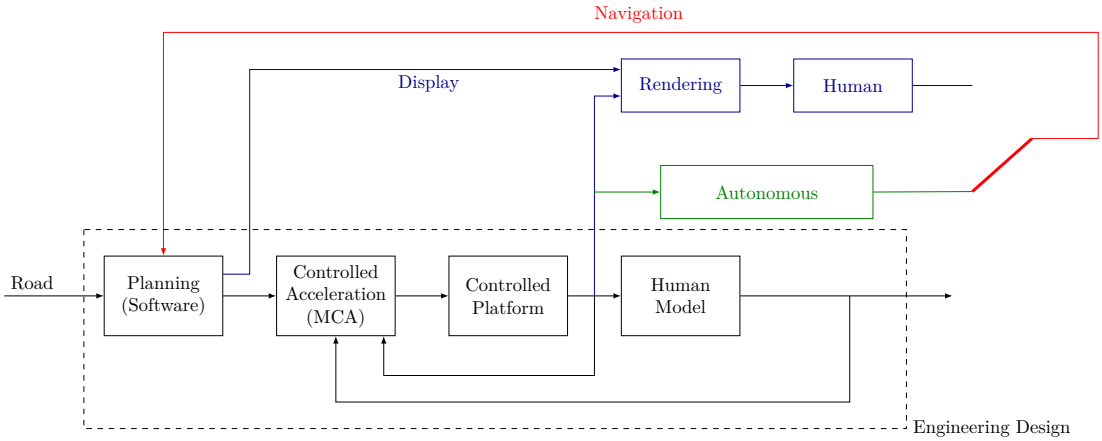

Figure 1.12: Scheme of the driving simulation operating in terms of control

Real-time Control

The real-time control of the simulation platform has to guarantee the restitution of the movement sensation in a restrictive area, thus it has to take into account:

1. The platform and driver dynamics, which concerns its mass, its degrees of freedom (translations and rotation in our case), the perception of the driver and the control and data processing architecture (electronic or numerical filters, communication protocol, ...) which may induces delays in the global response.

2. The physical limitations of the simulator (rail dimensions, mechanical resistance of the actuators, energy consumption). The strategies of limits avoidance are generally known to be more conservative.

3. The computation time of control signal to be send to the simulator has to be lower than the sampling time period of the real-time architecture.

In this thesis we aim to contribute to the enhancement of these 3 items by using a model-based and optimization based approach.

The Chapter 2 is dedicated to the Motion Cueing Algorithm (MCA) which is the module transforming the acceleration profiles into position and tilt references to be followed by the platform. The material contributes to the open literature by presenting a state of art of the different strategies developed in the last decades.

The Chapter 3 brings the first novelties proposed by the current research work, two MPC-based approaches:

- the first takes into account the nonlinearities caused by the rotational movements that are generally neglected. It resumes the results published in [Soyer et al., 2021a] and proposes a control architecture in chain allowing the compensation of the translational dynamics by the rotational one and conversely.

- the second proposes a constant delay compensation caused by simulator inertia. The main contribution is the use of an alleviated MPC controller from a computational point of view allowing the delay handling all by guaranteeing the stability and feasibility of the procedure [Soyer et al., 2021b].

In the Chapter 4, we focus on the constrained tracking problem underlying the dynamic behaviour of the simulator. From the theoretical point of view, our aim is to prove that Interpolation-Based Control (IBC) procedure is a suitable design framework. Is worth to be noted that IBC for tracking was an open problem at the beginning of the current research project and its extension to this particularly important practical control problem has been addressed in [Soyer et al., 2020].

Based on these theoretical contributions, we adapt the technique to the driving simulation along the results presented in [Soyer et al., 2020a].

The Chapter 5 addresses the problem of the computation time in a broader scope. By assuming it depends on the complexity of the constrained set in the optimal control framework, we propose to relax the property of positive invariance to a periodic notion which has the fundamental advantage of relaxing the constraints on the set description and consequently open the way for simpler constraints in the MPC design. However, the generalization from classical positive invariance to the periodic notions is shown to be a particularly rich topic and we established a formal classification by pointing to notions of *weak* and *strong* invariance as potential candidate formulations. Eventually, our aim is to provide a formal framework allowing to manipulate these concepts and construct practically invariant candidate set for the class of autonomous dynamics [Soyer et al., 2020b]. Then we extend these results to the controlled systems by presenting MPC-based and IBC-based procedures [Soyer et al., 2020a] in order to use these new feature in real-time driving simulator controllers [Soyer et al., 2020b].

1.3.2 Publications

Conference papers

- **M. Soyer**, S. Olaru and Z. Fang, "Interpolation Based Control for reference tracking under constraints", 2020 European Control Conference, Saint Petersburg, Russia, 2020, pp. 855-860
- **M. Soyer**, S. Olaru, Z. Fang, "From constraint satisfactions to periodic positive invariance for discrete-time systems", Conference on Decision and Control, Dec 2020, Jeju Island, South Korea
- **M. Soyer**, S. Olaru, Z. Fang, D. Wautier and A. Kemeny, "Interpolation-Based MCA for acceleration rendering", Driving Simulation & VR Conference, Sep 2020, Antibes, France
- **M. Soyer**, S. Olaru and Z. Fang, "A novel Motion Cueing Algorithm based on real-time optimization and periodic invariant sets", Conference on Control Technology and Applications, Aug 2020, Montréal, Canada.
- **M. Soyer**, S. Olaru, K. Ampountolas, S. Scialanga and Z. Fang, "Periodic Set Invariance as a Tool for Constrained Reference Tracking", 2020 IFAC WC, Berlin, Germany
- **M.Soyer**, S.Olaru, and Z.Fang. "Motion Cueing Control Design Based On A Nonlinear Mpc Algorithm." IFAC-PapersOnLine 54.6 (2021): 341-346.

- **M.Soyer**, S. Olaru and Z.Fang. "MPC delay compensation based on maximal controllable sets for real-time driving simulators." 2021 25th International Conference on System Theory, Control and Computing (ICSTCC). IEEE, 2021.

Article (submitted)

- **M.Soyer**, S. Olaru, C. E. T. Dórea and E. Kofman. "From relaxed constraint satisfaction to p-invariance of sets", Transaction on Automatic Control.

Poster

- **M. Soyer**, S. Olaru, Z. Fang, D. Wautier and A. Kemeny," Periodic Invariance in MCA to address conservativeness and computation time issues", Driving Simulation & VR Conference, Sep 2021, Munich, Germany

Chapter 2

MCA: modeling and state of art

Contents

2.1 Control Architecture

2.1.1 MCA in driving simulator control architecture

The control architecture of ULTIMATE-like dynamic driving simulator is depicted on Figure 2.1, by underlying its three main components:

1. **The generation of the acceleration signal** to be rendered to the driver in the cabin is done by means of an independent software which builds on the driving scenarios and the available information on the driver behavior (experience, age, mass, sport style, leisure, etc). *Ex: If the driver turns the steering wheel on a roundabout, the value of the lateral acceleration signal during the simulation has to increase and the amplitude and the profile will consider the trajectory before the roundabout and the path to be followed at the exit*

2. **MCA (Motion Cueing Algorithm)**: processes the resultant acceleration signal towards admissible position and tilt reference for the electro-mechanical dynamic platform. The present research work is is dedicated to a great extent to this component.

3. **Electro-mechanical controlled structure** designed by a supplier of car manufacturers. As the platform is made of rails and hexapod, its dynamics are nonlinear (a full description can be found in [Elloumi, 2006]) and then the low level control can be designed using a feedback linearization technique [Dagdelen et al., 2009].

Consequently, MCA is an intermediary function driving indirectly the actuators in order to follow an acceleration profile trajectory in a reduced workspace.

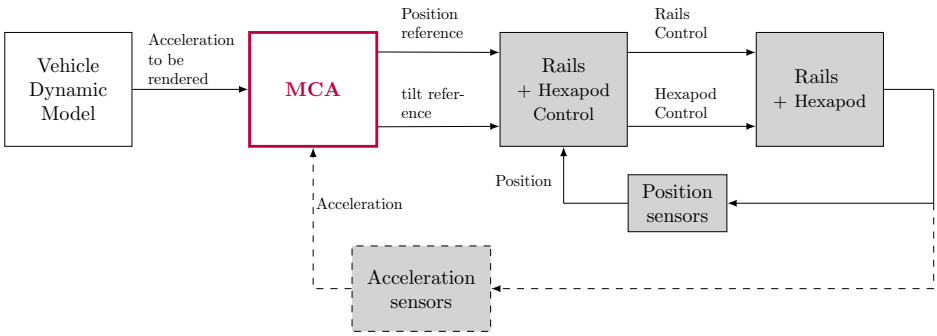

Figure 2.1: Scheme of the control architecture of dynamic driving simulator, grey filled elements are supplier's component that cannot be designed from our level.

Remark. *The MCA control design can follow an open loop architecture approach or a closed loop approach, depending of industrial choices.*

Given the complex structure of the platform and the interplay in between the different blocks, a series of assumptions and model reductions need to be considered in order to concentrate on the essential phenomena to be mastered within the admissible limitations.

Assumption 2.1

In the following, we consider, unless otherwise specified, that the internal control of the MCA transforms an acceleration into position and tilt angle profiles by taking into account physical electro-mechanical structure with a perfect response with respect to a sensing in the range of tens of milliseconds. In other words, the mechanical inertia is negligible and there are no delays due to the communication protocol.

2.1.2 Workspace and actuators

The workspace is an area of less than 30 square meters where the platform moves laterally and longitudinally thanks to electric rails that delimit the moving area as depicted on Figure 2.2. For obvious security reasons, this area is not accessible and the only access point is the platform at the docking position.

The actuators of the hexapods are electrical pistons operated simultaneously so as to tilt the platform as illustrated on Figure 2.3.

The MCA block which we are focusing in aims to provide the best inputs signal to those actuators in order to reproduce as accurately as possible the equilibrium feeling for the driver in a virtual, dynamically moving scene.

2.1.3 Sensed acceleration/specific force

The human body senses acceleration excitations thanks to the vestibular system which is detailed in section 2.4.1 within a biological perspective. However, we can estimate the acceleration felt by the driver in the cabin thanks to classical mechanical modelling tools.

Given the configuration of the driving simulator depicted on Figure.2.3 with a tilt angle θ and a longitudinal rail acceleration a_{lin}, the acceleration felt by the driver can be deduced from the mechanical fundamental principle of dynamics in the non-inertial frame of the driver on his longitudinal x axis:

$$ma_{driver} = -ma_{lin}\cos(\theta) - mg\sin(\theta) \tag{2.1}$$

Then the specific acceleration felt by the driver is:

$$a_{felt} = a_{lin}\cos(\theta) + g\sin(\theta) \tag{2.2}$$

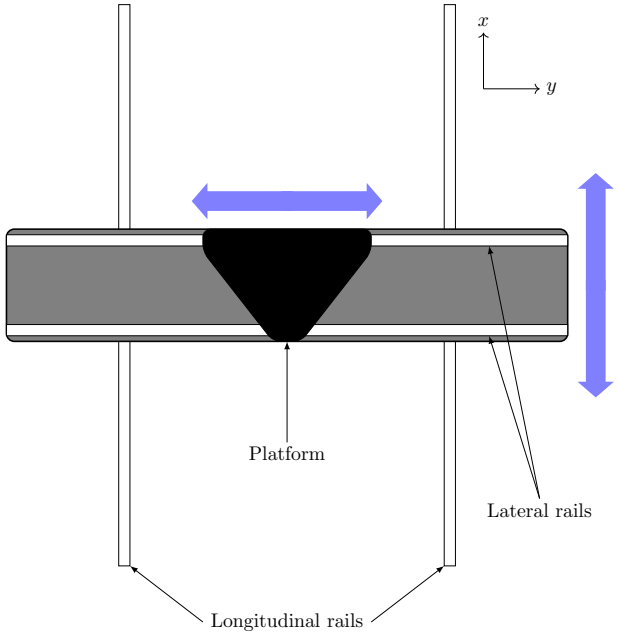

Figure 2.2: Workspace view from above

Assumption 2.2

Unlike the aircraft simulators, the tilt angle is generally small enough such that the output was generally linearized considering $\cos(\theta) \approx 1$ and $\sin(\theta) \approx \theta$. In the present manuscript this will admit the working assumption, and whenever it is of interest to exploit or to mitigate the nonlinear behaviour, this will be specified.

Assumption 2.3

While the previous studies count on the fact that specific force in (2.2) is filtered by the inner ear, we will consider this component explicitly in the subsequent developments.

2.2 MCA: function and modeling

In this section, we focus on filter-based design for the philosophical and historical perspective, then we address the model predictive control. In the end we introduce

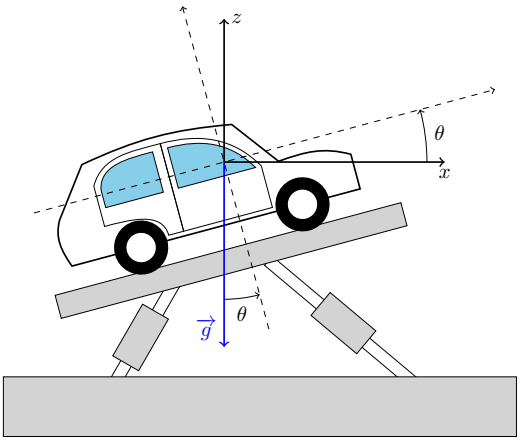

Figure 2.3: Scheme of side-view of dynamic driving simulator

derived techniques.

There exists several ways to model and design the MCA as per the technological choices of manufacturers. The first MCA were based on filtering while the most recent one are designed through optimal control framework or neural networks.

2.2.1 Filter-based design

The filter based MCA has been used in the development of flight simulation in [Schmidt and Conrad, 1970]. The main philosophy of this strategy is illustrated on Figure 2.4 and can be summarized as follows: the translation acceleration signal is splitted in two components

- one enabled by a High Pass filter (HP) corresponding to the quick movements. *Ex: Turning the steering wheel.*
- one enabled by a low pass filter (LP) corresponding to slow movement. *Ex: Keeping the steering wheel at a specific angle.*

Quick movements are directly integrated into position reference for the dynamic platform while slow movement are rendered by **tilt coordination**. In parallel, the quick angular accelerations of the car are also directly restituted by tiliting.

The filters are designed in the way of avoiding the actuators limitations. Practically, the cut-off frequencies and gain of the filters are selected such that the responses to a constant input satisfy the constraints. Table 2.2.1 inspired from [Fang and Kemeny, 2012a] presents the transfer functions associated to each channel and the range of their parameters.

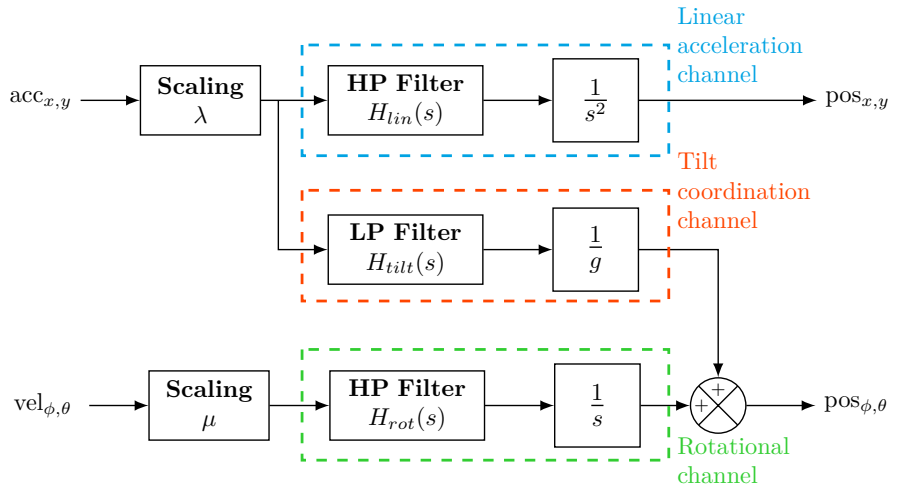

Figure 2.4: Scheme of the filter-based MCA

These parameters have to be tuned according to the simulator architecture, the driver characteristics and the scenario. The linear acceleration channel can be modelled by the filter $H_{lin}(s)\frac{1}{s^2}$, the final position according to a step input signal of amplitude a_{ref} is:

$$\lim_{t\to\infty} p(t) = \lim_{s\to 0} sH_{lin}(s)\frac{1}{s^2}\frac{a_{ref}}{s} = \frac{k_1}{\omega_{n1}^2}a_{ref} \tag{2.3}$$

So, the platform tends to remain to its final position which can have a negative impact on the overall behaviour when another increase of acceleration occurs in the same direction at a later stage. A component can be added to the channel such that the final value of the platform achieves a zero steady-state error which is equivalent to the neutral position. This technique is known as **washout** and will be also consider in other types of MCA. We propose here a generic definition:

Definition 2.1: Washout

The **washout** process is a component of the MCA which tends to get the platform back to its neutral position after a sequence of piecewise constant accelerations.

Thus, the channel $H_{lin,w}\frac{1}{s^2}$ has a neutral final position when excited by a unit step

Channel	Transfer function	Parameters
Linear acceleration	$H_{lin}(s) = \frac{k_1 s^2}{(s^2 + 2\xi_1\omega_{n1}s + \omega_{n1}^2)}$	$\omega_{n1} \in [2.5, 4] rad/s$ $\xi_1 \in [1, 1.4]$
Linear acceleration (washout)	$H_{lin,w}(s) = \frac{k_1 s^3}{(s^2 + 2\xi_1\omega_{n1}s + \omega_{n1}^2)(s + \omega_w)}$	$\omega_{n1} \in [2.5, 4] rad/s$ $\xi_1 \in [1, 1.4]$ $\omega_w \in [0.1, 0.5] rad/s$
Tilt Coordination	$H_{tilt}(s) = \frac{k_2 s^2}{(s^2 + 2\xi_2\omega_{n2}s + \omega_{n2}^2)}$	$\omega_{n2} \in [0.65, 2.5] rad/s$ $\xi_2 = 1$
Rotation	$H_{rot}(s) = \frac{k_3 s}{s + \omega_b}$	$\omega_b = 0.2$

Table 2.1: Transfer functions and values of cut-off frequencies and damping coefficients [Fang and Kemeny, 2012a]

input:

$$\lim_{t\to\infty} p(t) = \lim_{s\to 0} sH_{lin,w}(s)\frac{1}{s^2}\frac{1}{s} = 0 \tag{2.4}$$

The main advantage of this filter-based strategy is its simple implementation and design. In comparison with the other techniques, there is no computational burden. However, this technique tends to be very conservative, which means that the workspace is not used at the boundary at its operating regime at least compared to its theoretical capabilities. Another main drawback of this strategy is the perception of nonexpected feelings known as **backlash effect**([Fang and Kemeny, 2014, Nehaoua et al., 2006]).

Definition 2.2: Backlash Effect

The backlash effect is the nonexpected acceleration feeling induced by the MCA misinterpretation of the acceleration decrease to 0 as a braking.

Example: Filter design and backlash effect

We propose in Table 2.2 values for the parameters:

H_{lin}	H_{tilt}
$\omega_{n1} = 0.9$, $k_1 = 1$, $\omega_w = 0.2$, $\xi_1 = 1.4$	$k_2 = 0.5$, $\omega_{n2} = 1.5$, $\xi_2 = 1$

Table 2.2: Parameters chosen for the filter-based design

Consider the rendering of a constant acceleration of 3m/s^2 and a brake until immobilization, Figure.2.5 depicts the response of each MCA channel as well as the global response. To fulfill the constraints, the reference signal has been reduced by 40% ($\lambda = 0.6$).

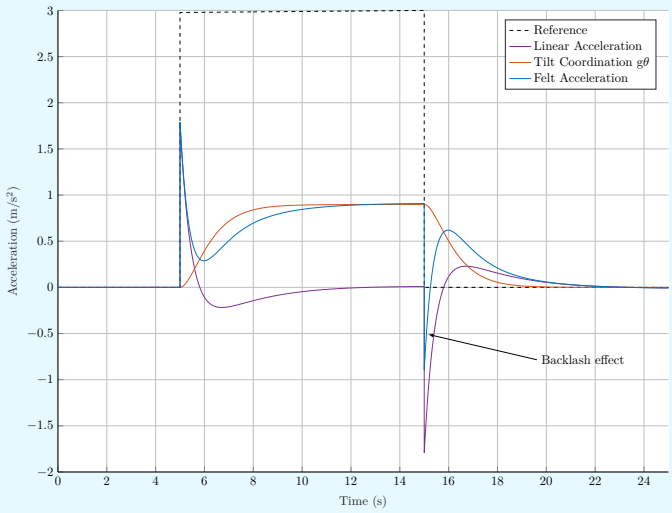

Figure 2.5: MCA response to a windowed acceleration signal

The backlash effect can be seen while braking, the linear acceleration tends to accelerate on the opposite side which make feel the driver nonexpected sensations.
We consider the scenario with the following features:

- We focus on the lateral movements.
- The first phase is an exaggerated slalom.
- The second phase is a great turn.

The scenario is a kind of worst case, the slalom phase tends to evaluate the rail response while the turn tests the tilt response. The specific force is represented on Figure 2.6.

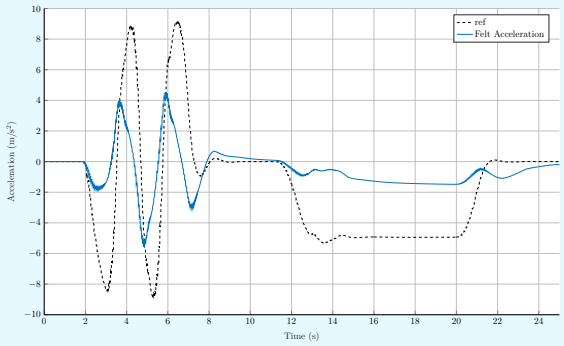

Figure 2.6: Acceleration felt by the driver as a function of time. While the length of the acceleration dynamical sequence is similar with the reference one, there is a delay in the sensed acceleration which may induce a discomfort at the driver side.

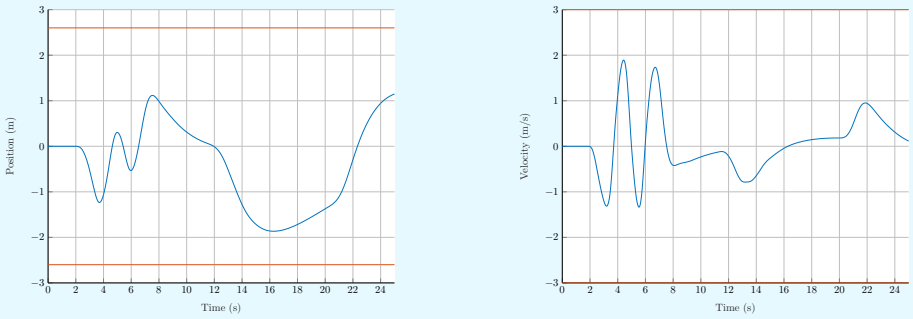

Figure 2.7: Left: lateral rail position, Right: lateral rail velocity. The limitations of the driving simulators are illustrated by the red line

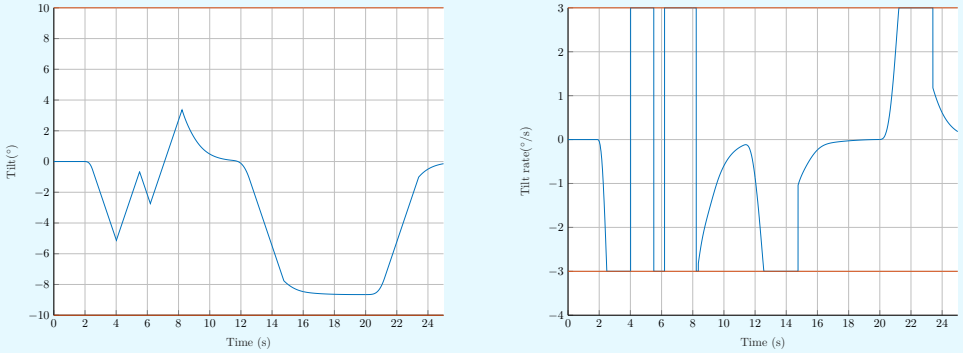

Figure 2.8: Left: roll angle, Right: roll tilt rate. The limitations of the driving simulators are illustrated by the red line

2.2.2 Model Predictive control design

As explained previously, once the inertia and the response dynamics of the platform are considered to be fast with respect to the acceleration sensing the MCA can be modeled as a chain of integrators under constraints (Figure 2.9). Here the state space is discrete for the real time requirements of the optimal control close loop implementation. Several choices can be done for the state space under

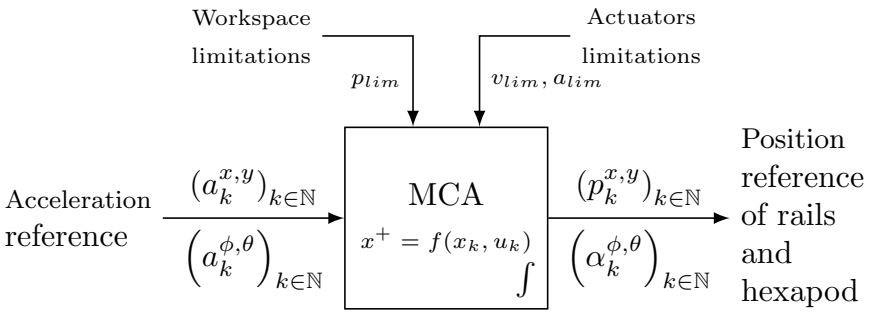

Figure 2.9: Scheme of MCA function

computational capacities.

4 states: Simple and computational attractive double integrator

If we consider the lateral rail dynamics, in the continuous-time framework, the position $p(t)$ can be retrieved from the acceleration signals:

$$\ddot{p}(t) = a(t) \tag{2.5}$$

Consequently, the continous state $x(t) = \begin{bmatrix} p(t) & \dot{p}(t) \end{bmatrix}^T$ verifies the linear differential equation:

$$\dot{x}(t) = \underbrace{\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}}_{A_c} x(t) + \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{B_c} a(t) \tag{2.6}$$

The solution of the differential equation initiated in t_0 is given by:

$$x(t) = e^{A_c(t-t_0)}x(t_0) + \int_{t_0}^{t} e^{A_c(t-\tau)}B_c a(\tau)d\tau \tag{2.7}$$

Since we aim to discretize the equation with a sampling time T_s, we consider the classical assumption of constant input between two time instants kT_s and $(k+1)T_s$

(implemented using a zero order hold):

$$\begin{aligned} x((k+1)T_s) &= e^{A_c T_s} x(kT_s) + \int_{kT_s}^{(k+1)T_s} e^{A_c[(k+1)T_s - \tau]} B a(kT_s) d\tau \\ x(k+1) &= e^{A_c T_s} x(k) + \int_0^{T_s} e^{A_c \xi} B d\xi a(k) \end{aligned} \tag{2.8}$$

Thanks to Definition A.3, and by noticing that A_c is nilpotent of order 2 ($A_c^2 = 0_2$):

$$x(k+1) = \begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} \frac{T_s^2}{2} \\ T_s \end{bmatrix} a(k) \tag{2.9}$$

By doing the same development for the tilt dynamics, we can obtain the final discrete-time state space model by setting the $x(k) = \begin{bmatrix} p(k) & v(k) & \theta(k) & \Omega(k) \end{bmatrix}^T$ and $u(k) = \begin{bmatrix} a(k) & \gamma(k) \end{bmatrix}^T$:

$$\begin{aligned} x(k+1) &= \begin{bmatrix} 1 & T_s & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T_s \\ 0 & 0 & 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} \frac{T_s^2}{2} & 0 \\ T_s & 0 \\ 0 & \frac{T_s^2}{2} \\ 0 & T_s \end{bmatrix} u(k) \\ y(k) &= \begin{bmatrix} 0 & 0 & g & 0 \end{bmatrix} x(k) + \begin{bmatrix} 1 & 0 \end{bmatrix} u(k) \end{aligned} \tag{2.10}$$

where $p(k)$, $v(k)$ and $a(k)$ are respectively the position, the speed and the acceleration of the platform w.r.t the rail while $\theta(k)$, $\Omega(k)$ and $\gamma(k)$ are respectively the tilt angle, angular velocity and acceleration.

6 states: inclusion of the jerk, triple integrator

By using the same type of construction, a 6 dimensional model using the triple integration of the jerk $j(t)$ can be obtained:

$$\dddot{p}(t) = j(t) \tag{2.11}$$

By setting the $x(k) = \begin{bmatrix} p(k) & v(k) & a(k) & \theta(k) & \Omega(k) & \gamma(k) \end{bmatrix}^T$ and $u(k) = \begin{bmatrix} j_r(k) & j_\theta(k) \end{bmatrix}^T$, then we have:

$$
\begin{aligned}
x(k+1) &= \begin{bmatrix} 1 & T_s & \frac{T_s^2}{2} & 0 & 0 & 0 \\ 0 & 1 & T_s & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & T_s & \frac{T_s^2}{2} \\ 0 & 0 & 0 & 0 & 1 & T_s \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} \frac{T_s^3}{6} & 0 \\ \frac{T_s^2}{2} & 0 \\ T_s & 0 \\ 0 & \frac{T_s^3}{6} \\ 0 & \frac{T_s^1}{2} \\ 0 & T_s \end{bmatrix} u(k) \\
y(k) &= \begin{bmatrix} 0 & 0 & 1 & g & 0 & 0 \end{bmatrix} x(k)
\end{aligned}
\tag{2.12}
$$

where we use the same notation than the previous section and $j_r(k)$ and $j_\theta(k)$ are the rail and hexapod jerks.

This model is more complex and implies a heavier computational burden but it allows to handle information about the jerk as independent input signal while the output is a linear function of the state all by preserving the controllability properties on all the components subject to constraints. In the following, we will use the the classical literal formulation:

$$
\begin{aligned}
x(k+1) &= Ax(k) + Bu(k) \\
y(k) &= Cx(k) + Du(k)
\end{aligned}
\tag{2.13}
$$

Assumption 2.4: Origin of the state space

The origin of the state space corresponds to the static position on the center scene for the platform. In the control perspective, the origin becomes an admissible point in the interior of the operational domain.

Constraints

- **Workspace constraints**: The platform is symmetric and we set the middle of the platform as the origin and the rest position:

$$
-p_{max} \leq p(k) \leq p_{max} \; \forall k \in \mathbb{N} \tag{2.14}
$$

- **actuators constraints**: The energy consumption and the material resistance of actuators implies a constraints on speed, acceleration and jerk.

$$
\begin{aligned}
-v_{max} \leq v(k) \leq v_{max} \\
-a_{max} \leq a(k) \leq a_{max} \\
-j_{max} \leq j(k) \leq j_{max}
\end{aligned}
\tag{2.15}
$$

Where the bounds p_{max}, v_{max}, a_{max}, j_{max} are strictly positive and the acceleration and jerk can be adapted to avoid important jerk and enhance the safety.

In the following, we will use the set framework to handle the constraints, thus the set of states constraints (respectively inputs constraints) will be denoted by $\mathcal{X}$ (respectively $\mathcal{U}$)

Assumption 2.5

In this work we will always refer to the constraints of the Renault's ULTIMATE driving simulator and to the perception thresholds of the vestibular system summarized in Table 2.3.

ULTIMATE			
Rails		Hexapod	
p_{max}	2.6 m	θ_{max}	$10°$
v_{max}	3 m/s	ω_{max}	$3°/s$
a_{max}	3 m/s^2	Ω_{max}	$15°/s^2$
$j_{r,max}$	600 m/s^3	$j_{\theta,max}$	$1000°/s^3$

Table 2.3: Physical limitations of the ULTIMATE driving simulator

Delays

There exists different sources of delays:

- *Communication protocol*: The control signals and measurements are broadcast in real-time through well known communication protocols (scheduling, TCP/IP, LAN, ...). We consider here a constant transmission delay [Fang et al., 2010].
- *Mechanical inertia*: Even if the simulator's dynamics is idealized in this thesis, there exists a phase delay due to the actuators motion. [Fang et al., 2010] proposes models for the actuators on the form:

$$G(s) = \frac{K}{(1 + T_1 s)} e^{-\tau_d s} \tag{2.16}$$

  However, we assume in this work the delays to be constant.
- *Prediction*: According to the reliability of the predicted trajectory, the positioning and tilt of the platform may not be optimal. The prediction issues are discussed in Section 2.4.1. Obviously, the control performances will be tributary to the quality of the predicted trajectory by the human.

Then the delayed system can be modeled as:

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k-d) \\ y(k) &= Cx(k) + Du(k-d) \end{aligned} \tag{2.17}$$

where $d \in \mathbb{N}$ is the constant delay.

On this part we focus on the receding horizon optimal control as a generic framework policies for the MCA. The principle of MPC is summarized below and illustrated on Fig.2.10 and relies on the implementation of the first part of the control sequence found as the result of the optimization:

$$\begin{aligned} \underset{(u(k),\ldots,u(k+N-1))}{\text{minimize}} \quad & J = \sum_{i=1}^{N} J_{k+i}(x(k),\ldots,x(k+i),u(k),\ldots,u(k+i-1)) \\ \text{subject to} \quad & x(k+1) = f(x(k),u(k)), \\ & (x(k+1),\ldots,x(k+N-1)) \in \mathcal{X}, \\ & (u(k),\ldots,u(k+N-1)) \in \mathcal{U}, \\ & x(k+N) \in \mathcal{X}_f \end{aligned} \tag{2.18}$$

Where J is the cost function to be minimized, $(J_k)_{k=1,\ldots,N}$ are the partial costs, $(\mathcal{X},\mathcal{U})$ are respectively the sets of state and inputs constraints associated to (2.14) and (2.15) and $\mathcal{X}_f$ is a positive invariant set. This main property is defined next.

Definition 2.3: Positive invariance

A set $\mathcal{C} \subset \mathcal{X}$ is said to be Positively Invariant w.r.t the autonomous system $x(k+1) = f(x(k))$ if for any initial state initial state $x(0) \in \mathcal{C}$ the state trajectory $x(1)$ remains in $\mathcal{C}$.

Performance Criteria

- **Tracking error**:

$$J_y(k) = \|y^{ref}(k) - y(k)\|_{q_y}^2 \tag{2.19}$$

  where y^{ref} is the acceleration reference given by the software and the main element toward the restitution of the movement in the simulator, y is the estimated acceleration and $q_y \in \mathbb{R}^+$ the weight coefficient associated to the tracking.

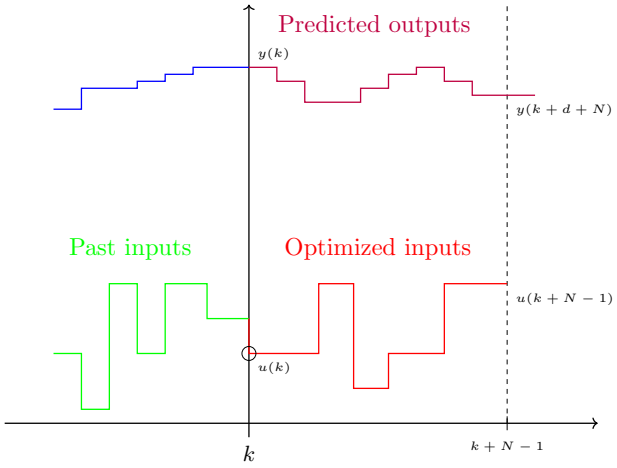

Figure 2.10: MPC operating: the controller finds an optimal sequence of inputs (in red) that implies the predicted output (in purple), only the first component of the sequence is applied.

- **Washout law**:

$$J_x(k) = \|x(k)\|^2_{Q_x} \tag{2.20}$$

where $Q_x \in \mathbb{R}^{n\times n}$ is symmetric positive semidefinite. This costs penalizes the geometrical distance from the origin, in such a way that in the absence of (acceleration) excitation the platform will spontaneously come back to its initial position. However this washout effect has to be transparent for the driver in the simulator. The movements are slow for the sensed motion but the washout effect is particularly important for a better use of the workspace on the long simulation scenarios..

- **Control weight**:

$$J_u(k) = \|u(k)\|^2_R \tag{2.21}$$

where $R \in \mathbb{R}^{m\times m}$ is symmetric positive semidefinite.

Given the performance criteria of the previous section and the constraints (2.14)

and (2.15), the MPC formulation becomes:

$$\begin{aligned}
\underset{(u(k),\ldots,u(k+N-1))}{\text{minimize}} \quad & \sum_{i=1}^{N} \|y^{ref}(k+i) - y(k+i)\|_{Q_y}^2 + \|x(k+i)\|_{Q_x}^2 \\
& + \|u(k+i)\|_R^2 + \|x(k+N)\|_P^2 \\
\text{subject to} & \\
x(k+i+1) = Ax(k+i) + Bu(k+i) \quad & \forall i \in \{0,\ldots,N-1\}, \\
y(k+i) = Cx(k+i) + Du(k+i) \quad & \forall i \in \{0,\ldots,N\}, \\
x(k+i) \in \mathcal{X} \quad & \forall i \in \{1,\ldots,N\}, \\
u(k+i) \in \mathcal{U} \quad & \forall i \in \{0,\ldots,N-1\}, \\
x(k+N) \in \mathcal{X}_f &
\end{aligned} \tag{2.22}$$

In the following we will consider implicitly that $\mathcal{X}_f$ is the maximal positively admissible set (*i.e* the largest admissible positive invariant set) with respect to a LQ controller (see Definition A.22)

Example: MPC-MCA

We consider the MPC controller (2.22) with the weights $N = 200$, $Q_y = 1000$, $Q_x = diag(100, 100, 1, 1)$ and $R = diag(1000, 100)$. The constraints are the ones given in Table 2.3. The scenario is the same as the one given in Example 2.2.1. The Figure 2.11 shows the specific force felt by the driver during the scenario and the Figure 2.12 depicts the states of the simulator. One can notice that this strategy is obviously better than the filter-based one in terms of performance because of the prediction and the constraint handling. However we will address the drawback in terms of implementation later.

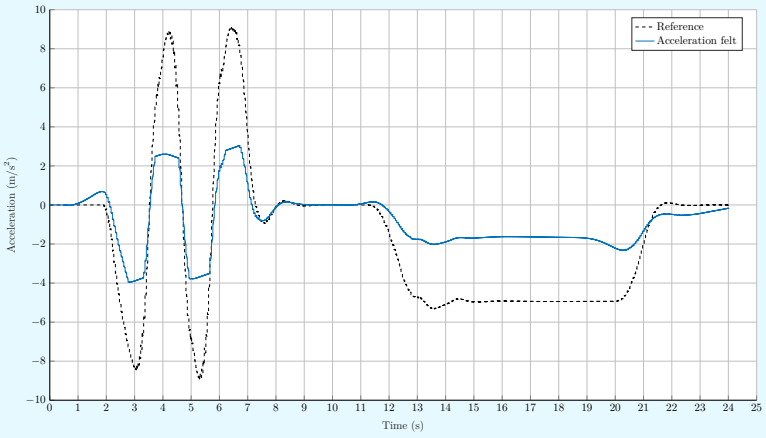

Figure 2.11: Acceleration felt by the driver

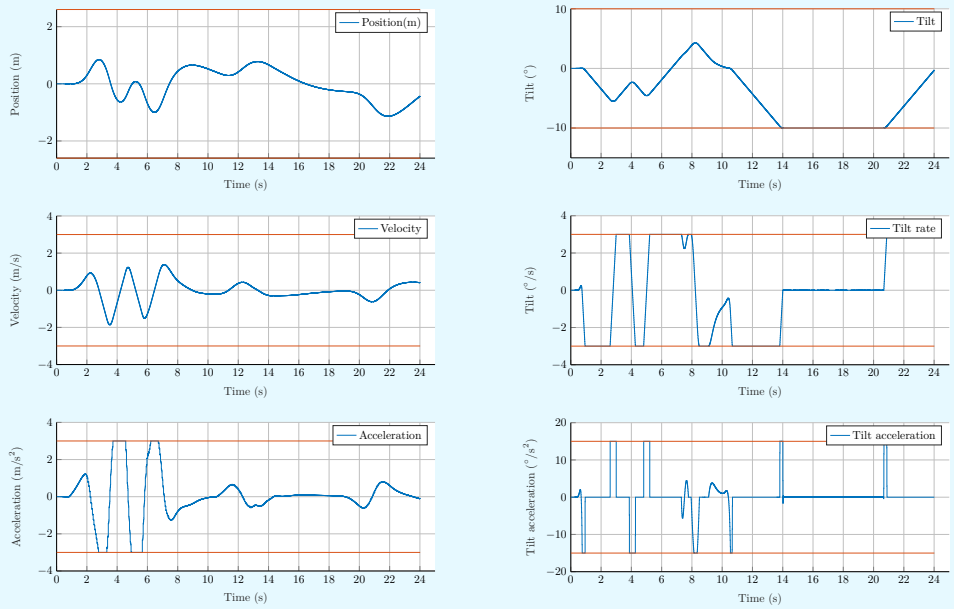

Figure 2.12: Left: Rails states (position, velocity and acceleration. Right: Hexapod states (tilt angle, tilt speed and tilt acceleration). The limitations are depicted by red lines.

2.3 Other types of MCA

2.3.1 Enhanced filter-based design: Adaptative filters

An alternative of filter-based MCA had been investigated in [Parrish et al., 1975], the main idea was the adaptation in real time of the filters parameters by optimizing a cost function representing the performance of the motion restitution. Consider the High-Pass filter of the linear acceleration channel (Figure.2.4) without washout component:

$$\frac{1}{s^2}H_{lin}(s) = \frac{P(s)}{A^{veh}(s)} = \frac{k_1 s^2}{(s^2 + 2\xi\omega_{n1}s + \omega_{n1}^2)} \tag{2.23}$$

where $P(s)$ and $A^{veh}(s)$ are respectively the Laplace transform of the position $p(t)$ and vehicle acceleration $a^{veh}(t)$. In the time domain, the associated differential equation is:

$$\ddot{p}(t) = k_1 a^{veh}(t) - 2\xi\omega_{n1}\dot{p}(t) - \omega_n^2 p(t) \tag{2.24}$$

Now, we consider the gain k_1 as an adaptative parameter denoted P_1. We also consider the modified rotational equation proposed in [Telban et al., 2000] using gains P_2 associated to tilt coordination and P_3 for the tracking of vehicle tilt rate:

$$\begin{aligned} \ddot{p}(t) &= P_1 a^{veh}(t) - 2\xi\omega_{n1}\dot{p}(t) - \omega_n^2 p(t) \\ \dot{\theta}(t) &= P_2 a^{veh}(t) + P_3 \omega^{veh}(t) \end{aligned} \tag{2.25}$$

Thanks to those relationships one can build a quadratic cost function involving the design gains which can thus be optimized:

$$\begin{aligned} J(P) = \frac{1}{2}\Big[& \overbrace{w_a(a^{veh} - \ddot{p})^2 + w_t(\omega^{veh} - \dot{\theta})^2}^{\text{Tracking errors}} + \overbrace{w_p p^2 + w_v \dot{p}^2 + w_\theta + w_r \dot{\theta}^2}^{\text{Washout}} \\ & + \underbrace{w_1(P_1 - P_{1,0})^2 + w_2(P_2 - P_{2,0})^2 + w_3(P_3 - P_{3,0})^2}_{\text{Parameters change penalty}} \Big] \end{aligned} \tag{2.26}$$

with $P = [P_1\, P_2\, P_3]^T$, $P_{1,0}$, $P_{2,0}$, $P_{3,0}$ are nominal gains and w_a, w_t, w_p, w_v, w_{theta}, w_r, w_1, w_2, w_3 are positive weights associated to each quadratic component of the cost function (2.26) which represents the trade-off between tracking errors (between vehicle movements and restitution) and parameters modifications, thus the controller changes the parameters only if it is useful.

Remark. *The cost function can be modified for the need of the designers, particularly the vestibular system dynamics (see section 2.4.1) can be added to the dynamics* (2.25) *as in [Ariel and Sivan, 1984] where the semicircular canals are considered.*

The cost function (2.26) is optimized thanks to the gradient descent which means the new parameters are chosen such that their variation is opposite and colinear to the gradient of J:

$$\dot{P}_i = -\alpha_i \frac{\partial J}{\partial P_i}(P), \qquad i \in \{1, 2, 3\} \tag{2.27}$$

where $\alpha_i > 0$ is the optimization step.

The main advantage of this technique is the addition of a degree of flexibility on the parameter tuning during the driving simulation. However, it should be said that the proof of stability guarantees for this time-varying dynamical system were missing and thus, without surprise, it is not used anymore with the exception of specific driving simulation applications. Moreover, aside the stability guarantees the potential improvements are counterbalanced by a non-negligible computational and implementation effort.

2.3.2 MPC with braking law

One of the most important features in MCA design is the efficient use of the workspace. As the simulator has a very strict range of movements, an acceleration of the vehicle may lead to a displacement of the platform that rapidly reaches the workspace limits. The mainstream solution is to reduce the acceleration signal to be rendered as we have seen in the filter-based design case. Thus one can expect, for example, a rendering of 60% of the acceleration profile instead of 100% by guaranteeing that workspace limits will not be hit. However the simulator movement becomes naturally and practically restricted to the neighborhood of the neutral position, in this case the MCA is said to be **conservative** which means the margin of displacement is high.

The other approach is the active constraints handling which can be provided by closed loop procedures and particularly by optimization-based controllers such as MPC. Theoretically, at each time step, the controller computes a control action that guarantee the constraints satisfaction and consequently the platform movements are strictly contained within the workspace limits. However, practically when a constraint in position is activated ($|p(k)| = p_{max}$), the only feasible solution is to accelerate toward the opposite direction (break) which implies motion sickness and can lead to severe conditions. This issue can be avoid thanks to the parameter tuning by increasing the weight on washout component or the prediction horizon. Indeed, a long prediction horizon implies a better simulator trajectory and a better constraints handling but the compromise is a higher computation time, potentially greater than the sampling time, more as we will show in section 2.4.1 the prediction is not always possible.

A compromise has been investigated in [Fang and Kemeny, 2012b] and [Fang and Kemeny, 2016] whose main idea is to smooth the braking deceleration before activating the limitations. At each step of the MPC-MCA, the following condition is checked:

$$|p(t) + v(t)T + u(t)\frac{T^2}{2}| \leq p_{max} \qquad i.e \qquad |p(t+T)| \leq p_{max} \tag{2.28}$$

where we use the previous notations ($p(t)$ is the position, $v(t)$ the velocity and $u(t)$ stands for acceleration) and $T > 0$ is a time horizon. As long as the condition (2.28) is valid, MPC operates normally but when the constraint is activated the following condition is imposed:

$$p(t) + v(t)T + u_b(t)\frac{T^2}{2} = p_{max} \tag{2.29}$$

and the associated Laplace transform of the position becomes:

$$P(s) = \frac{\left[\frac{p_0}{p_{max}}s^2 + \frac{1}{p_{max}}(v_0 + 2\xi\omega_n p_0) + \omega_n^2\right]}{s^2 + 2\xi\omega_n s + \omega_n^2} . \frac{p_{max}}{s} \tag{2.30}$$

where (p_0, v_0) correspond to the state of the platform when (2.29) holds, $\omega_n = \frac{\sqrt{2}}{T}$ and $\xi = \frac{\sqrt{2}}{2}$ which means the position of the platform operates as a second order time invariant oscillator excited by a step input with an amplitude of p_{max} and $p(t)$ may exceed p_{max} due to the overshoot. To overtake this issue [Fang and Kemeny, 2012b] proposed a modified condition:

$$p(t) + c_v v(t) T + c_u u_b(t) \frac{T^2}{2} = p_{max} \tag{2.31}$$

with parameters $c_v > 0$ and $c_u > 0$, thus the Laplace transform of the $p(t)$ has the same formulation as (2.30) but with

$$\omega_n = \frac{1}{T}\sqrt{\frac{2}{c_u}} \qquad \text{and} \qquad \xi = \frac{c_v}{\sqrt{2c_u}} \tag{2.32}$$

Then the new parameters c_v and c_u can be chosen such that the system becomes an overdamping one ($\xi > 1$). In the time domain, the position is given by:

$$p(t) = p_{max}\mathbb{1}(t) + A_1 e^{-\beta_1 \xi \omega_n t} + A_2 e^{-\beta_1 \xi \omega_n t} \tag{2.33}$$

$$\begin{aligned} with \quad A_1 &= \frac{v_0 + \xi\omega_n(2p_0 - 2p_{max} - \beta_1 p_0 + \beta_1 p_{max})}{\xi\omega_n(\beta_2 - \beta_1)} \\ A_2 &= -\frac{v_0 + \xi\omega_n(-2p_0 + 2p_{max} + \beta_1 p_0 - \beta_1 p_{max})}{\xi\omega_n(\beta_2 - \beta_1)} \end{aligned} \tag{2.34}$$

One can verify the steady states values of position and velocity:

$$\lim_{t\to\infty} p(t) = p_{max} \qquad \text{and} \qquad \lim_{t\to\infty} v(t) = 0 \tag{2.35}$$

which means the platform practically brakes before the rail stops and without changing direction. The resulting input can be computed as:

$$u_b(t) = C_1 e^{-\beta_1 \xi \omega_n t} + C_2 e^{-\beta_2 \xi \omega_n t} \tag{2.36}$$

$$\text{with} \quad \beta_1 = 1 - \sqrt{1 - \tfrac{1}{\xi^2}} \qquad \beta_2 = 1 - \sqrt{1 + \tfrac{1}{\xi^2}}$$

$$\begin{aligned} C_1 &= \frac{\omega_n(v_0 - 2\beta_1\xi^2 v_0 - \beta_1\xi\omega_n p_0 + \beta_1\xi\omega_n p_{max})}{\xi(\beta_1 - \beta_2)} \\ C_2 &= \frac{\omega_n(v_0 - 2\beta_2\xi^2 v_0 - \beta_2\xi\omega_n p_0 + \beta_2\xi\omega_n p_{max})}{\xi(\beta_1 - \beta_2)} \end{aligned} \tag{2.37}$$

A speed limitation can be taken into account by considering an alternative asymptotic law:

$$u_{vlim}(t) = \frac{1}{T_v} e^{-\frac{t}{T_v}} \tag{2.38}$$

with $T_v > 0$. Thus the input signal provided to the simulator is:

$$u(t) = \begin{cases} \max\{u_b(t), u_{vlim}(t), -u_{max}\} \text{ if } p(t) > 0 \\ \min\{u_b(t), -u_{vlim}(t), u_{max}\} \text{ if } p(t) < 0 \end{cases} \tag{2.39}$$

2.3.3 Neural networks

More recently, the progress in artificial intelligence and machine learning fields has inspired the actual development of MCA based on neural networks. We succinctly present here two approaches:

1. [Rengifo et al., 2018] proposes to replace the optimization algorithm of the MPC-MCA by a Recurrent Neural Network (RNN) in order to reduce the computation time. Learning of the acceleration profiles can lead undoubtly to good performances for scenarios that have been already simulated and for which good state space occupation and performance can be ensured. However, on a larger scale, the approach can behave poorly for scenarios with poor information extraction from the available scenarios.

2. [Koyuncu et al., 2020] proposes to imitate an optimal control based MCA with an infinite horizon by using a neural network trained by data collected thanks to drivers in a static simulator. The resulting NN-MCA may be used in real time online dynamic simulator, but the same fundamental limitations as above can be expected and need to be faced with additional components.

In both cases the neural network emulates an optimal control structure or a part of it which needs an upstream tuning of the parameters. Those alternatives are promising for the performance enhancement and their low computation time but the need for a model-based control design with certifiable performances is clear even in these approaches.

2.4 Issues and open problems in MCA design

We have seen in this chapter several ways to structure a MCA, from filters to optimization based architectures. The restitution of inertial sensations in a restricted workspace naturally imposes to engineers to drop a complete restitution and impose the use of a partial one by making relaxation choices in the design. We summarize next the issues encountered in the design of MCA, we propose a classification in two categories of open problems:

- Those referring to human perception or behavior and the ways to take them into account in MCA design.

- Those referring to the control itself *i.e* the tuning of the parameters, the computation time and delay.

2.4.1 Human factors

Perception model

The perception of movements is provided by the contribution of vision and equilibrioception. Those two functionalities are both stimulated in the driving simulator. The vision is stimulated by graphical elements displayed on the screen and the response of the software with respect to the driver's actions. The equilibrium is stimulated through the vestibular system by the movement of the plateform, consequently, we focus on this system in the following.

There exists a difference between the effective acceleration of the driver and the acceleration effectively sensed. This gap is caused by the proper dynamic of the vestibular system located in the inner ear, its anatomy being illustrated on Figure.2.13 The organs responsible for informing the nervous system of the presence of an acceleration are explicited below:

- **The otolithic system** is responsible for the detection of the linear movements of the head. It is made of *utricle* which detects horizontal acceleration and *saccule* which detects vertical ones.

- **The semicircular canals** detects the rotations around their own axis.

One of the main issues for aeronautic and aerospace industry during the second part of the XX^{th} century was the mathematical modelisation of the dynamics of the vestibular system. The work of [Meiry, 1965] in 1965 was the first giving models of both otoliths and semicircular canals in terms of transfer function using identification and experimental results. The reviews [Asadi et al., 2016] and [Houck et al., 2005] provides a history of the modelling of otoliths and semicircular canals dynamics some of these results being reported in table 2.4.

From a mechanical point of view, the modelisation of otoliths is close to a a mass-spring system, which is relevant with respect to the anatomy of the otolithic system. Thus we can generalized the transfer function as:

$$H_{oto}(s) = \frac{\hat{f}(s)}{f(s)} = k_{oto}\frac{(1+\tau_a s)}{(1+\tau_l s)(1+\tau_s s)} \tag{2.40}$$

where τ_a is the afferent time constant (representing the phase delay of the neural system), τ_l is the long time constant and τ_s is the short time constant.

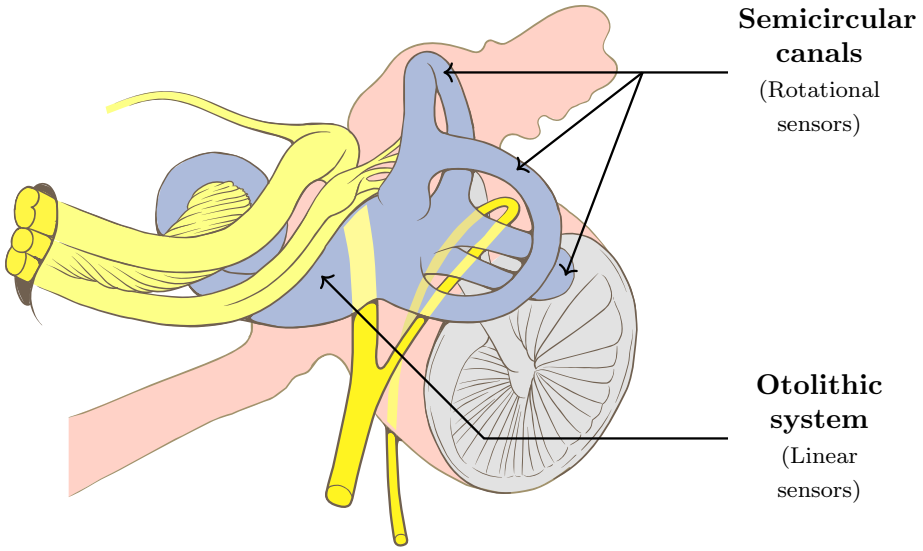

Figure 2.13: Anatomy of the vestibular system (inner ear). Credits: Patrick J. Lynch, medical illustrator; C. Carl Jaffe, MD, cardiologist, Creative Commons Attribution 2.5 License 2006; https://creativecommons.org/licenses/by/2.5/deed.fr, the original illustration has been modified with captions.

On the other hand the semicircular canals dynamics can be seen as a torsion pendulum represented by the following transfer function:

$$H_{scc}(s) = \frac{\hat{\omega}(s)}{\omega(s)} = k_{scc}\frac{\tau_a}{(1+\tau_a s)}\frac{\tau_l s^2(1+\tau_d s)}{(1+\tau_l s)(1+\tau_s s)} \tag{2.41}$$

where τ_a is the time constant of the adaptation dynamics τ_d is the afferent time constant (representing the phase delay of the the neural system), τ_l, τ_s are the long time constant and short time constants.

Prediction of the trajectory

In the previous section we considered a completely known vehicle acceleration profile but in practice such an information is not necessarily known in advance, at least not for a large time-window in the future. The acceleration trajectory to be tracked is available at the current moment in time and its extrapolation is subject to important uncertainties due to poor predictability of human reactions in relationship with the scene and the motion cueing algorithm. In control related terms, this means that the future trajectory $\{a^{ref}(k+1|k), a^{ref}(k+2|k), a^{ref}(k+3|k), ...\}$ can be made available but it cannot be granted to be receded at the next

Otoliths models		
Authors	**Model**	**Remark**
[Meiry, 1965]	$\frac{\hat{v}(s)}{a(s)} = \frac{K}{(1+10s)(1+0.66s)}$	$\hat{v}$: subjective velocity a: linear acceleration K: Gain (not measured) Contribution: Identification
[Young and Meiry, 1968]	$\frac{\hat{f}(s)}{f(s)} = \frac{K(1+\tau_a s)}{(\tau_L s+1)(\tau_s+1)} = \frac{0.4(13.2s+1)}{(5.33s+1)(0.66+1)}$	$\hat{f}$: sensed specific force f:stimulus of specific force Contribution:Correction of [Meiry, 1965] to add a neural contribution term
[Ormsby, 1974]	$\frac{E\{\hat{f}(s)\}}{f(s)} = 1.5\frac{(s+0.076)}{(s+0.19)(s+1.5)}$	$E\{.\}$ is the expectation of a random variable using a Wiener-Hopf method
[Houck et al., 2005]	$\frac{\hat{f}(s)}{f(s)} = 0.4\frac{(10s+1)}{(5s+1)(0.016s+1)}$	Modification of the small time constant
Semicircular models		
Authors	**Model**	**Remark**
[Meiry, 1965]	$\frac{\hat{\omega}}{\Omega} = \frac{7}{(7s+1)(0.1s+1)}$	$\hat{\omega}$: subjective tilt velocity Ω: angular acceleration Parameter identification
[Fernandez and Goldberg, 1971]	$\frac{\hat{\Omega}(s)}{\Omega(s)} = \frac{80}{(1+80s)}\frac{(1+0.49s)}{(1.57s)(1+0.003s)}$	$\hat{\Omega}$: sensed tilt acceleration Ω:angular acceleration Adding of a phase lead term
[Ormsby and Young, 1977]	$\frac{\hat{\omega}}{\omega} = \frac{540s^2}{(18s+1)(30s+1)}$	$\hat{\omega}$: sensed tilt velocity ω: tilt velocity Short time constant neglected
[Houck et al., 2005]	$\frac{\hat{\omega}(s)}{\Omega(s)} = \frac{5.73\times 80(1+0.06s)}{(1+80s)(1.573s)(1+0.05s)}$	

Table 2.4: Models of otoliths and semicircular canals

time instant (i.e $a^{ref}(k+j|k+i)$ may consistently differ from $a^{ref}(k+j|k+i+1)$ for $j > i+1 > 0$). The accurate tracking of a particular sequence can turn to a poor performance in case of trajectory update from the MCA. We can enumerate 3 main cases:

1. **No interaction with the driver**: the driver in the simulator is subject to a predefined scenario and the trajectory of the platform is planned offline. It might be seen as not a challenging framework due to the absence of the human in the loop, but it turns to be a particularly useful framework for all the test which corresponds to an autonomous driving scenario. The motion sickness need to be avoided in order to facilitate the homologation of the simulated driving scenarios in terms of autonomous driving and retrieve useful information on scenarios that cannot be tested on the circuit.

2. **The trajectory is globally known**: The driver has precise instructions during the whole session: the speed to be tracked, the braking phases, the turns, etc... Consequently, the acceleration profile resulting from the driver's commands may differ from the one calculated offline.

3. **"Free ride"**: The driver interacts with the simulator freely, so the reference trajectory is unknown and is hardly predictable.

In the first two cases, the acceleration reference can be estimated thanks to the trajectory prediction software and the MCA can be fine-tuned by adding prepositioning of the plateform for example which correspond to an optimization of the initial conditions.

In the third case, the trajectory is more difficult to predict because of its high dependence to the driver's personal behavior. To give an order of magnitude, we consider a turn of radius R, the lateral acceleration felt by the driver is proportional to the square of the longitudinal speed of the vehicle:

$$a_{lat} = \frac{v_{veh}^2}{R} \tag{2.42}$$

If a driver comes into the turn with a speed of 50km/h (13.9 m/s), he will feel a lateral acceleration 3 times higher than a more cautious driver who brakes to 30km/h (8.3m/s) before arriving to the turn. In practice, approximations of the reference trajectory are actually used in MPC, we present here two typical methods:

1. **Constant reference**: At sampling time k, the N next prediction reference value are set to the value at time k.

$$y^p(k+i) = y^{ref}(k) \ \forall i \in \{1, \dots, N\} \tag{2.43}$$

where $y^p(.)$ is the prediction sent to the MPC-MCA and N the prediction horizon. This approximation is very easy to set up but the implicit assumption about the driver's behavior do not use any relevant information about the reality. The Figure.2.14 shows how the prediction can be far from the real dynamics of the vehicle.

2. **Taylor's expansion** An easy way to add information to the prediction is to take into account the value of the derivatives of the real reference at each instant k:

$$y^p(k+i) = y^{ref}(k) + i\frac{dy^{ref}}{dt}(k) + \frac{i^2}{2}\frac{d^2y^{ref}}{dt^2}(k) + \dots \ \forall i \in \{1, \dots, N\} \quad (2.44)$$

Practically only the first and second order approximations are used, the Figure.2.4.1 depicts what can happen by using the first order approximation: the predicted trajectory does not take into account the inflexions or the saturation phenomenon.

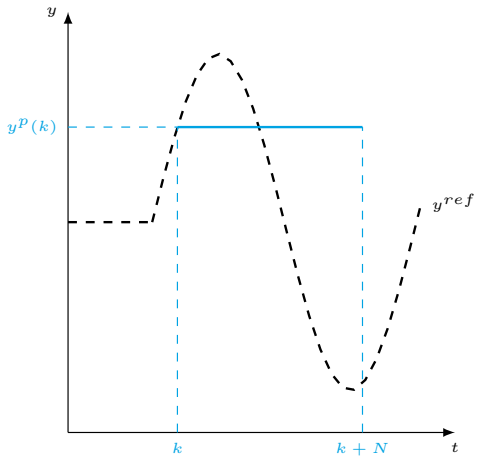

Figure 2.14: Prediction using a constant approximation

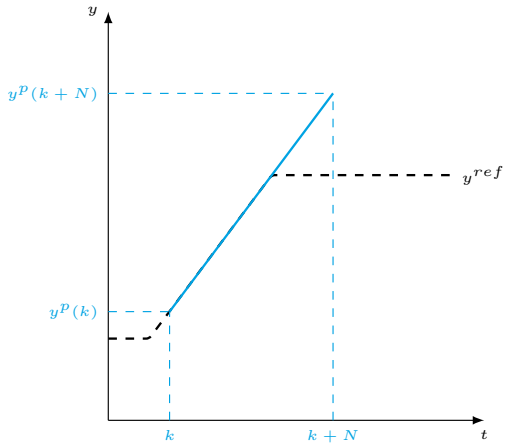

Figure 2.15: Prediction using a first order approximation

Very recent works aims to overcome the issue of prediction by using more information or by modelling the driver behavior:

- [Mohammadi et al., 2016] proposes a neural network which give a prediction of the future trajectory by processing a finite number of past inputs
- [Lamprecht et al., 2021] modelled the human behavior as an optimal controller that provide relevant future trajectories.

2.4.2 Motion cueing issues

Parameter Tuning

As we have seen in the section.2.2.2, the cost function needs a tuning of the weights such that there is a trade-off between tracking, washout and inputs amplitudes. However, many other components may be added to this function such as

- **The dynamics of the vestibular system** whose state space model can be derived from (2.40) for the otoliths and (2.41) for the saccules. Thus, (2.40) represents the following differential equation considering $f(t) = a + g\theta(t) = u_{lin}(t) + gu_{rot}(t)$:

$$\begin{aligned}\ddot{\hat{f}}(t) + \frac{\tau_l + \tau_s}{\tau_l\tau_s}\dot{\hat{f}}(t) + \frac{1}{\tau_l\tau_s}\hat{f}(t) = \frac{k_{oto}}{\tau_l\tau_s}u_{lin}(t) + \frac{k_{oto}\tau_a}{\tau_l\tau_s}\dot{u_{lin}}(t) \\ + \frac{k_{oto}g}{\tau_l\tau_s}\iint u_{rot}(t) + \frac{k_{oto}\tau_a g}{\tau_l\tau_s}\int u_{rot}(t)\end{aligned} \tag{2.45}$$

  which lead to a state space representation:

$$\begin{aligned}\dot{x}_{oto}(t) &= \begin{bmatrix} -\frac{\tau_l+\tau_s}{\tau_l\tau_s} & 1 & 0 & 0 \\ -\frac{1}{\tau_l\tau_s} & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} x_{oto}(t) + \begin{bmatrix} \frac{k_{oto}\tau_a}{\tau_l\tau_s} & 0 \\ \frac{k_{oto}}{\tau_l\tau_s} & 0 \\ 0 & \frac{k_{oto}\tau_a g}{\tau_l\tau_s} \\ 0 & \frac{k_{oto}g}{\tau_l\tau_s} \end{bmatrix} u(t) \\ \hat{f}(t) &= \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} x_{oto}(t)\end{aligned} \tag{2.46}$$

  In the same way, the equation (2.41) represents:

$$\begin{aligned}\dddot{\hat{\omega}}(t) + T_2\ddot{\hat{\omega}}(t) + T_1\dot{\hat{\omega}}(t) + T_0\hat{\omega}(t) = \frac{k_{scc}}{\tau_s}\dot{u}_{rot}(t) + \frac{k_{scc}\tau_d}{\tau_s}\ddot{u}_{rot}(t) \\ T_0 = \frac{1}{\tau_a\tau_l\tau_s}, \qquad T_1 = \frac{\tau_a + \tau_l + \tau_s}{\tau_a\tau_l\tau_s}, \qquad T_2 = \frac{\tau_a\tau_l + \tau_a\tau_s + \tau_l\tau_s}{\tau_a\tau_l\tau_s}\end{aligned} \tag{2.47}$$

  whose state space representation is:

$$\begin{aligned}\dot{x}_{scc}(t) &= \begin{bmatrix} -T_2 & 1 & 0 \\ -T_1 & 0 & 1 \\ -T_0 & 0 & 0 \end{bmatrix} x_{scc}(t) + \begin{bmatrix} 0 & \frac{k_{scc}\tau_d}{\tau_s} \\ 0 & \frac{k_{scc}}{\tau_s} \\ 0 & 0 \end{bmatrix} u(t) \\ \hat{\omega}(t) &= \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} x_{scc}(t)\end{aligned} \tag{2.48}$$

- **The prepositionning** *i.e* a tracking term driving the platform to a specific position. The feature can be used when the acceleration profile to render is well known.

- **The tracking of the vehicle angle**: during acceleration and braking phases the vehicle has an own tilt angle and rotational speed (independently of the tilt coordination technique) that can be restituted. It is the role of the channel rotational channel of the filter-based structure (Figure 2.4).By means of a cost function, the tracking term can be given by:

$$J_{\theta,veh}(k) = \|\theta^{veh,ref}(k) - \theta(k)\|^2_{q_{\theta,veh}} + \|\omega^{veh,ref}(k) - \omega(k)\|^2_{q_{\omega,veh}} \tag{2.49}$$

While better performance is expected from the use of the most information as possible in the cost function, the increasing number of weights may imply a complex relationship between all the components and the MCA becomes hard to tune. Moreover, It may be important to give a high weight on the washout component in order to avoid the rail stops which consequently increase the workspace use margin.

Computation time

Even if the problem is nonexistant in the filter-based design framework, it is a main barrier for the optimization based architecture. Indeed, the real-time structure imposes the execution time to be in the sampling time range. Since the execution speed of the optimization solver can be less than this limitation.

Delay compensation

As we have seen in Section 2.2.2, there exists different sources of delays. Their different kinds make the compensation being difficult. This latter depends on the reliability of the prediction as well as the MCA design.

Chapter 3

Enhanced MPC for MCA

Contents

In this chapter we focus on enhancements on the design of MPC based MCA starting from the structure presented in Section 2. In this respect, we propose two alternatives handling nonlinearity and delay :

1. The first one takes into account the nonlinearities caused by the projections on driver's frame axis during the modeling steps of Section.2.2. In other words, we avoid the linearization of the output all by handling the constraints. This approach may lead to an increase of the tilt amplitude. We propose here two approaches, each prioritising one of the internal dynamics.

2. The second one attempts to compensate the inertial delays explicited in the last chapter. The main idea is to find a lighter design (from a computational point of view) than existing techniques.

3.1 Inclusion of nonlinearities in MPC-MCA

The acceleration felt by the driver is a nonlinear combination of each type of actuators contributions on a given direction (lateral or longitudinal). Most approaches use linearized models for the MCA design and focus on the tuning of the cost function in order to account for the motion sickness as we have seen in Section.2.2. A different way to enhance the performances, through the use of a nonlinear constraints-handling approach will be considered next along two strategies:

1. The rail action is privileged and the tilt compensates the difference between the acceleration provided by rails and the expected specific force.

2. The tilt action is privileged and the rail compensate the tracking error.

The main idea from the design point of view is to separate the nonlinear model from the linear subsystem and thus handle a manageable complexity in the constrained optimization framework offered by the MPC strategy. Based on the mathematical nonlinear model and relative to these approaches, we compare their performance and computational load in view of real-time implementation.

Nonlinear MCA model

Given the configuration of the driving simulator depicted on Fig.2.3 with a tilt angle θ and a longitudinal rail acceleration a_{lin}, we consider the following state-space model with 3 states and 2 inputs:

$$x(k+1) = \begin{bmatrix} p(k) + T_s v(k) + \frac{T_s^2}{2} u_l(k) \\ v(k) + T_s u_l(k) \\ \theta(k) + T_s u_r(k) \end{bmatrix} = f(x(k), u(k)) \tag{3.1}$$

$$y(k) = g \sin(\theta(k)) + u_l(k) \cos(\theta(k)) = g(x(k), u(k))$$

where T_s is the sampling time, $x(k) = [p(k), v(k), \theta(k)]^\top$ denotes the state vector containing position, speed and tilt angle, $u(k) = [u_l(k), u_r(k)]^\top$ denotes the input vector containing linear acceleration (a_{lin} in (2.1) and (2.2)) and tilt rate. In other words, the rail dynamics is represented by a double integration of an acceleration input while the rotational dynamics is a single integrator controlled in tilt velocity.

Remark. *We recall here that the model is a choice among higher dimensional ones, the generic strategies presented next being applicable for them (see Section 2.2.2).*

For the dynamics (3.1), the MPC problem can be expressed as follow:

$$\begin{aligned}
\underset{(u(k),\ldots,u(k+N-1))}{\text{minimize}} \quad & \sum_{i=1}^{N} \|y^{ref}(k+i) - y(k+i)\|_{Q_y}^2 + \|x(k+i)\|_{Q_x}^2 \\
& + \|u(k+i)\|_R^2 + \|x(k+N)\|_P^2 \\
\text{subject to} & \\
x(k+i+1) = f(x(k+i), u(k+i)) \quad & \forall i \in \{0,\ldots,N-1\}, \\
y(k+i) = g(x(k+i), u(k+i)) \quad & \forall i \in \{0,\ldots,N\}, \\
x(k+i) \in \mathcal{X} \quad & \forall i \in \{1,\ldots,N\}, \\
u(k+i) \in \mathcal{U} \quad & \forall i \in \{0,\ldots,N-1\}, \\
x(k+N) \in \mathcal{X}_f &
\end{aligned} \tag{3.2}$$

This latter formulation has several drawbacks in view of real-time implementation, particularly it handles a 3 dimensional state space (or higher) and the nonlinear cost function is eventually nonconvex as function of the tilt angle range). Moreover, there are two distinct dynamics in the state space:

- The linear dynamics of the rails
- The nonlinear dynamics of the tilt

while the receding cost function mixes on a common prediction horizon the effect of two dynamics which prevents the decoupling.

Let us discuss two compensation mechanisms:

1. L-NL (Linear-NonLinear): Compensation of the linear component by the nonlinear one (Figure 3.1).

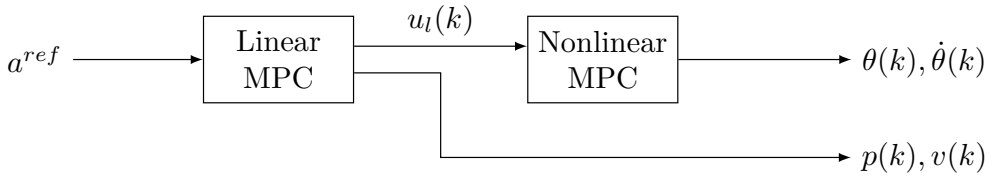

Figure 3.1: L-NL principle

2. NL-L (NonLinear-Linear): Compensation of the Nonlinear component by the linear one (Figure 3.2).

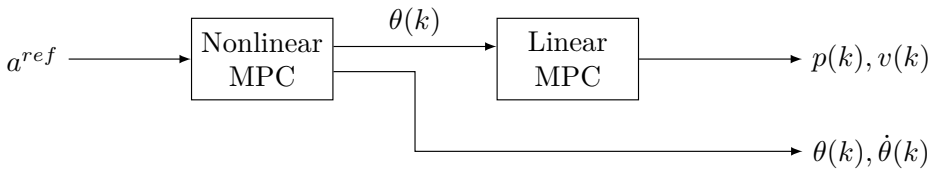

Figure 3.2: NL-L principle

In the following we will denote the linear rail dynamics as:

$$x_l(k+1) = \underbrace{\begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix}}_{A_l} x_l(k) + \underbrace{\begin{bmatrix} \frac{T_s^2}{2} \\ T_s \end{bmatrix}}_{B_l} u_l(k) \tag{3.3}$$

where $x_l(k) = [p(k)\ v(k)]^T$ is the state of the platform relatively to the rail and $u_l(k)$ is the acceleration input. Similarly, we denote the rotational dynamics as:

$$\theta(k+1) = \theta(k) + T_s u_r(k) \tag{3.4}$$

where $u_r(k)$ is the tilt velocity of the platform.

3.1.1 Linear-NonLinear (L-NL)

This formulation focuses on the rail movement, assumes the feasible current tilt angle $\hat{\theta}(k)$ as known and solves a linear MPC with the prediction horizon N_L in order to deliver a maximal contribution of the linear acceleration with respect to the given reference:

$$\begin{aligned} \underset{(u_l(k),\ldots,u_l(k+N_L-1))}{\text{minimize}} \quad & \sum_{i=1}^{N_L} \|y^{ref}(k+i) - (g\sin(\hat{\theta}(k)) + u_l(k+i))cos(\hat{\theta}(k))\|^2_{q_{y,L}} \\ & + \|x_l(k+i)\|^2_{Q_x} + \|u_l(k+i)\|^2_{R_L} \\ \text{subject to} & \\ x_l(k+1) = A_l x_l(k) + B_l u_l(k) \quad & \forall i \in \{0,\ldots,N_L-1\}, \\ x_l(k+i) \in \mathcal{X}_l \quad & \forall i \in \{0,\ldots,N_L-1\}, \\ u_l(k+i) \in \mathcal{U}_l \quad & \forall i \in \{0,\ldots,N_L-1\}, \\ x_l(k+N_L) \in \mathcal{X}_{f,l} & \end{aligned} \tag{3.5}$$

Where $\mathcal{X}_{f,l}$ is a positively invariant set constructed from the system constraints $\mathcal{X}_l$ and $\mathcal{U}_l$ and N_L is the prediction horizon of this problem. It is important to

mention that N_L is chosen independently, for this sub-system and can be adjusted to enlarge the domain of attraction and fulfill the computational constraints

The compensation is done by a nonlinear MPC with a prediction horizon $N_R \leq N_L$ using the optimal predicted control action $(u_l^*(k), \ldots, u_l^*(k+N_L-1))$ for (3.5):

$$\begin{aligned}
&\underset{\substack{(u_r(k),\ldots,\\ u_r(k+N_R-1))}}{\text{minimize}} \quad \sum_{i=1}^{N_R} \|(y^{ref}(k+i) - \hat{u}_l(k+i)cos(\theta(k+i))) - g\sin(\theta(k+i)))\|^2_{q_{y,R}} \\
&\qquad\qquad + \|u_r(k+i)\|^2_{R_R} \\
&\text{subject to} \\
&\theta(k+1) = \theta(k) + T_s u_r(k) \qquad \forall i \in \{0,\ldots,N_R-1\}, \\
&\theta(k+i) \in \mathcal{X}_R \qquad \forall i \in \{0,\ldots,N_R-1\}, \\
&u_r(k+i) \in \mathcal{U}_R \qquad \forall i \in \{0,\ldots,N_R-1\}
\end{aligned} \tag{3.6}$$

The operating of L-NL is summarized on Figure.3.3, the "Sel" block corresponds to the selection of the first input *i.e* the matrix [1 0 ... 0].

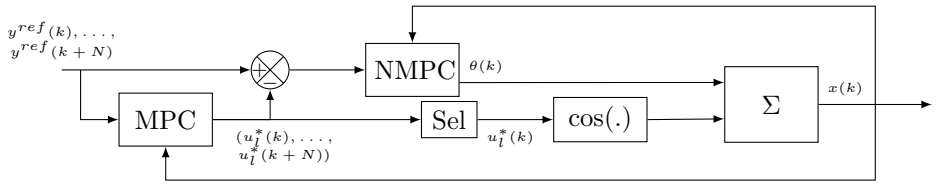

Figure 3.3: Block diagram of L-NL control structure

In the end, the resulting angle $\theta(k)$ is used in the next time instant $k+1$ such as: $\hat{\theta}(k+1) = \theta(k)$. We now analyse the performance through an example.

Example: L-NL MPC

We consider the study case described in Section 2 using the horizons $N_L = 100$ and $N_R = 30$. Thus, the rail dynamics predicts farther than the rotational one. We also weight the two cost functions terms of (3.5) and (3.6) such that $q_{y,L} = 10^6$, $Q_x = diag(50000, 800)$, $R_L = 10^6$, $q_{y,R} = 1000$ and $R_R = 1$. For the sake of analysis, this MCA has been implemented thanks to YALMIP interface of MATLAB, the optimization (3.5) is solved with *quadprog* MATLAB solver while (3.6) is solved with $fmincon$ function. The Figure.3.4 depicts the two components of the specific force and the comparison between this latter with the acceleration reference. The evolution of the states through time are represented on Figure.3.5 while the Figure.3.6 depicts the input signals evolution.

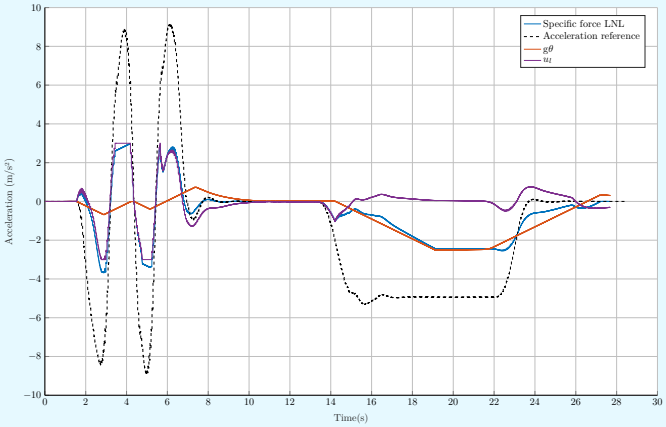

Figure 3.4: Specific force felt by the driver with respect to L-NL scheme

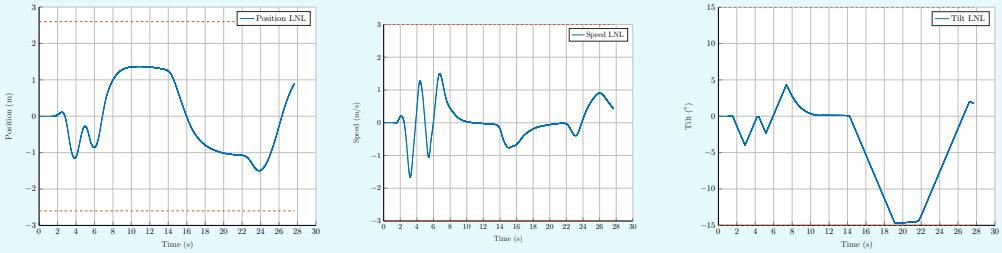

Figure 3.5: Position of rails (left), Speed of rail (center), tilt angle (right) in function of time

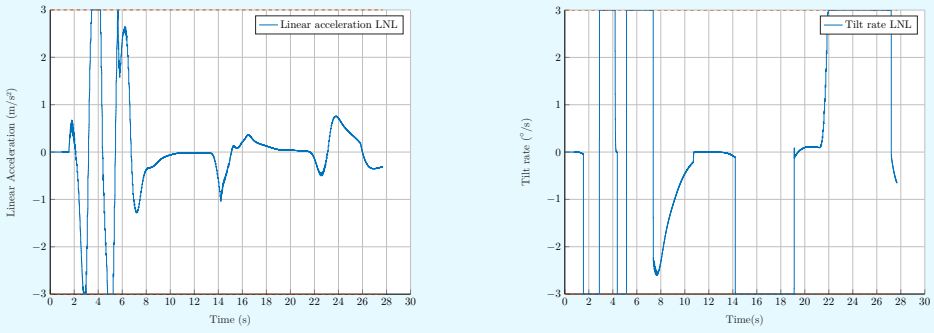

Figure 3.6: Inputs as function of time: Linear acceleration (right) and tilt rate (left)

We can notice the compensation mechanism around 7s, while the specific force decreases faster than the reference the tilt angle still increases in order to compensate the difference. In this particular example, the performance are poor because of the weak horizon on tilt, the turns are not well anticipated during the resolution of (3.6). The Figure.3.7 represents the computation time at each time instant which practically has to be less than the sampling time (here 8ms), here it can be higher because of the use of YALMIP. We depict it here as a term of comparison for the next approach.

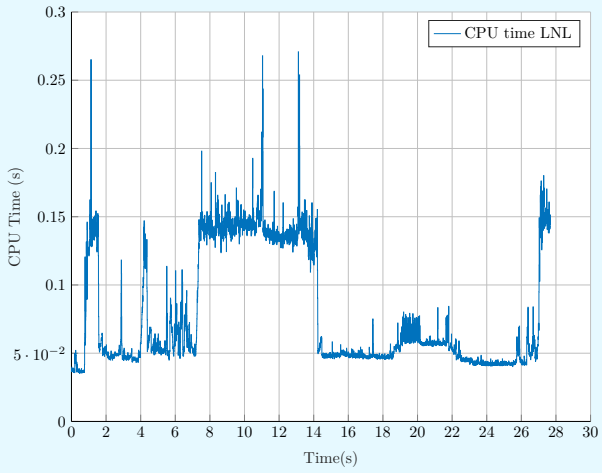

Figure 3.7: Computation time for L-NL

3.1.2 NonLinear-Linear (NL-L)

In this approach, the rotational component is privileged thanks to the projection of gravitational field on the lateral axis. The nonlinear optimization problem is similar to (3.6) except that there is no estimation of linear acceleration available:

$$\begin{aligned} \underset{(u_r(k),\dots,u_r(k+N_R-1))}{\text{minimize}} \quad & \sum_{i=1}^{N_R} \|(y^{ref}(k+i) - g\sin(\theta(k+i)))\|_{q_y}^2 + \|u_r(k+i)\|_R^2 \\ \text{subject to} \quad & \\ \theta(k+1) = \theta(k) + T_s u_r(k) \qquad & \forall i \in \{0,\dots,N_R-1\}, \\ \theta(k+1) \in \mathcal{X}_R \qquad & \forall i \in \{0,\dots,N_R-1\}, \\ u_r(k) \in \mathcal{U}_R \qquad & \forall i \in \{0,\dots,N_R-1\} \end{aligned} \tag{3.7}$$

The notations are the same as L-NL formulation.

The compensation is then done with a linear MPC framework over a prediction horizon $N_L \leq N_R$ using the available prediction on tilt angle $(\theta^*(k),\dots,\theta^*(k+N_L-1)$ solution of the optimization (3.7):

$$
\begin{aligned}
\underset{(u_l(k),\ldots,u_l(k+N_L-1))}{\text{minimize}} \quad & \sum_{i=1}^{N_L} \|y^{ref}(k+i) - (g\sin(\theta^*(k+i)) + u_l(k+i))\|_{q_y}^2 \\
& + \|x_l(k+i)\|_{Q_x}^2 + \|u_l(k+i)\|_R^2 \\
\text{subject to} & \\
& x_l(k+1) = Ax_l(k) + Bu_l(k), \\
x_l(k+i) \in \mathcal{X}_l \qquad & \forall i \in \{0,\ldots,N_L-1\}, \\
u_l(k+i) \in \mathcal{U}_l \qquad & \forall i \in \{0,\ldots,N_L-1\}, \\
& x_l(k+N_L) \in \mathcal{X}_{f,l}
\end{aligned}
\tag{3.8}
$$

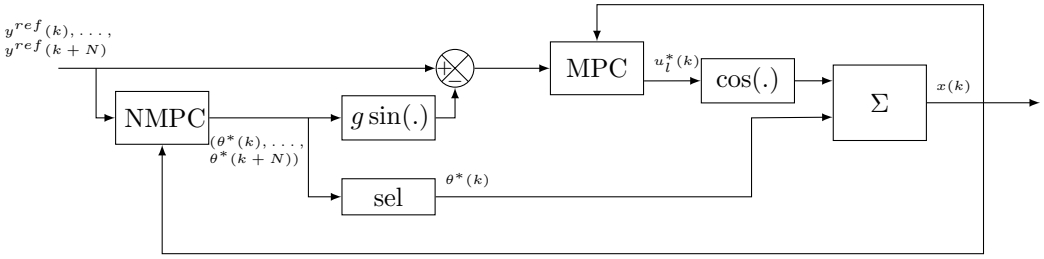

Figure 3.8: Block diagram of NL-L operation mode

Example: NL-L MPC

We consider the study case described in Section 2 using the horizons $N_L = 30$ and $N_R = 30$. Thus, the hexapod dynamics predicts at least on the same horizon as the rail. We also weight the two cost functions terms of (3.7) and (3.5) such that $q_{y,L} = 10^6$, $Q_x = diag(1,1)$, $R_L = 5.10^6$, $q_{y,R} = 1000$ and $R_R = 1$. Using the same implementation as previously. The Figure.3.9 depicts the two components of the specific force and the comparison between this latter with the acceleration reference. The evolution of the states through time are represented on Figure.3.10 while the Figure.3.11 depicts the input signals evolution.

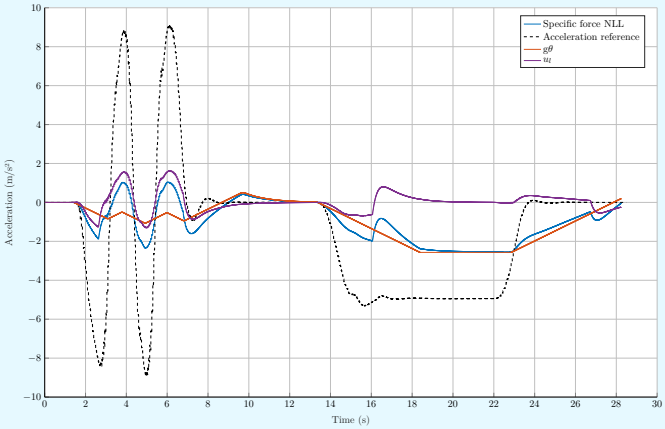

Figure 3.9: Specific force felt by the driver with respect to NL-L scheme

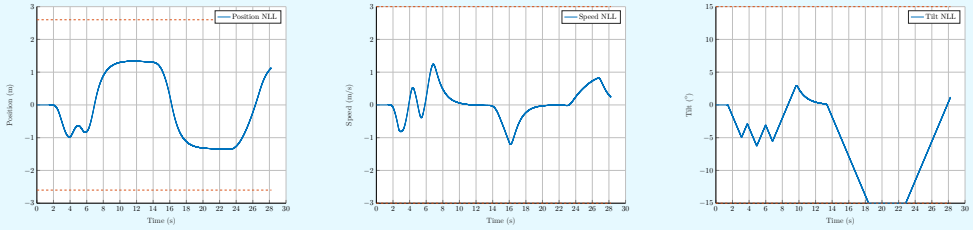

Figure 3.10: Position of rails (left), Speed of rail (center), tilt angle (right) in function of time

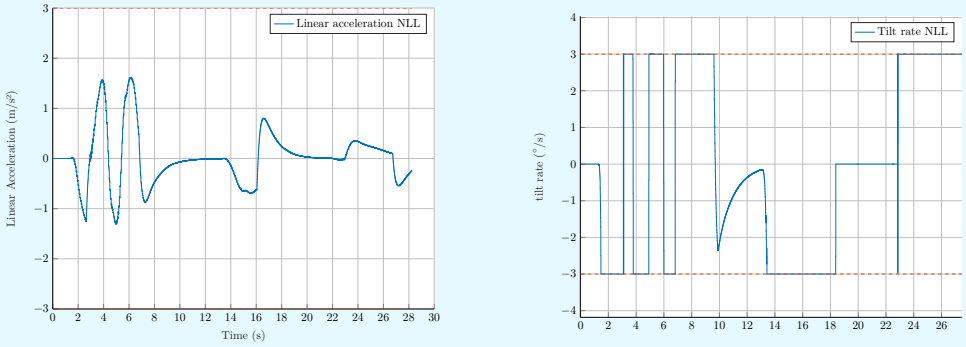

Figure 3.11: Inputs in function of time: Linear acceleration (right) and tilt rate (left)

The Figure.3.7 represents the computation time at each time instant.

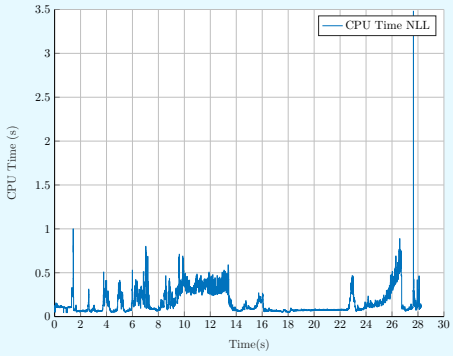

Figure 3.12: Computation time for NL-L

Comparison L-NL/NL-L

The Fig. 3.13 shows the mean computation time of optimization problems as a function of prediction horizons N_L and N_R. As expected, small horizons imply a faster resolution of optimization but we observe that globally the NL-L approach (left part of the first bisector) adds more computational burden than L-NL (right part). Consequently, it is more interesting for a computational point of view to implement a L-NL MCA, moreover by choosing this approach we can increase the performance with the prediction horizons on the linear part, with clear advantages for the feasible domain perspective.

3.1.3 Conclusion

In this subsection, we proposed an analysis of two design approaches for nonlinear MPC-based Motion Cueing Algorithm in view of real-time implementation with the goal of decoupling the linear from the nonlinear part.

The weight given to a part compared to the other (L-NL or NL-L) underlines their asymmetric role in the control design. Even if the nonlinear contribution (and thus the rotation) is higher for the tracking of acceleration, its computational cost tends to increase faster as a consequence of a prediction horizon that has to be larger or equal to the linear MPC dedicated to the rail system. Consequently the proposed design tends to privilege the contribution of rails, the increase of linear horizon stabilizes the response by a better management of rail displacement completed by the tilt.

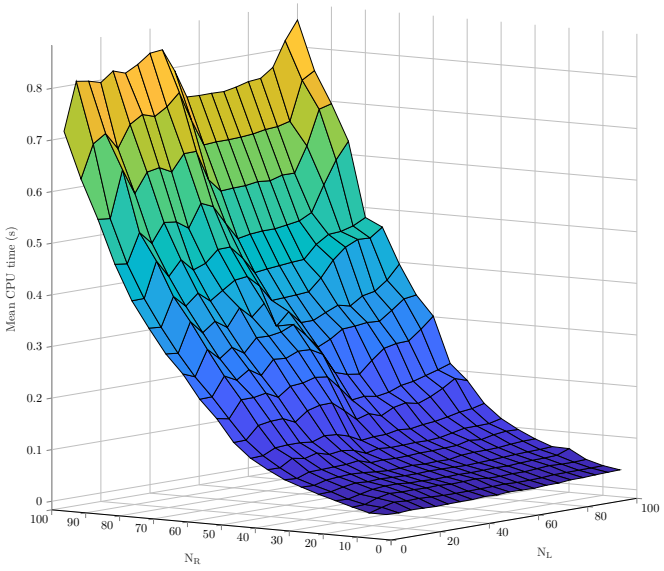

Figure 3.13: Mean CPU time function of linear prediction horizon N_L and rotational prediction horizon N_R

3.2 MPC with delay compensation

Aside the constraint handling and the computational limitations of MPC-MCA, the inherent delays are particularly difficult to handle in the control design with direct implications in the motion sickness phenomena. The goal of this section is to propose a control design which prevents the driver from feeling unease, by dealing with the time-delay from the design stage in the predictive control framework.

The classical approach for the delay compensation consists in translating the problem into an extended state space representation using past control actions and then design a MPC controller for the undelayed resulting model [Laraba et al., 2017]. However, this approach is faces computational limitations particularly for a real-time implementation due to the curse of dimensionality which emerges with the extended state space and become more difficult to overcome when the delay increases. Most publications deal with this issue by proposing set-invariance based perspective [Laraba et al., 2016, Laraba et al., 2017] or investigate an approach close to the Smith predictor philosophy [Santos et al., 2010].

In this work, we propose an MPC design based on the explicit use of the maximal controllable set or its approximations (see Definition 3.2) for delayed linear systems in the MPC.

3.2.1 State space model and delay compensation

In the presence of time-delay the discrete-time linearized model given by the state-space representation using 3 states and 2 inputs becomes:

$$x(k+1) = \begin{bmatrix} 1 & T_s & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} \frac{T_s^2}{2} & 0 \\ T_s & 0 \\ 0 & T_s \end{bmatrix} u(k-d) \tag{3.9}$$

$$y(k) = \begin{bmatrix} 0 & 0 & g \end{bmatrix} x(k) + \begin{bmatrix} 1 & 0 \end{bmatrix} u(k-d)$$

Assumption 3.1: Delays

In this section, we assume the both actuation channels are associated to the same delay. This can always be enforced by considering d to be the upper bound of delays on the respective actuation channels. In practice, even if they present slightly differences, the delays are in the range (50-100 ms).

The Figure 3.14 depicts the operating principles for the control system (3.9). The objective is to provide an accurate tracking of the acceleration signal.

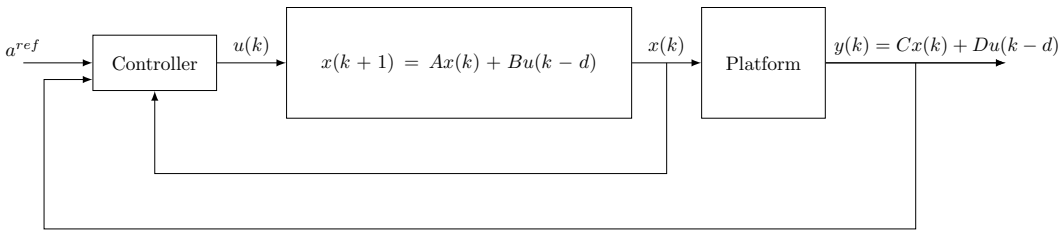

Figure 3.14: Block diagram of the control structure operating

Taking into account also the time-delay particularity of the dynamical model, this last remark imposes recursive feasibility requirements exclusively based on the current state and the previous control inputs.

Compensation with extended state space

The classical approach for the constrained control design for time-delay systems is the predictive control (MPC) using an extended state-space model. This state-space prediction model includes the past control actions in the extended state

vector

$$\xi(k) = \begin{bmatrix} x(k) & u(k-d) & \dots & u(k-1) \end{bmatrix}^\intercal .$$

With this artefact, the extended dynamics become:

$$\begin{aligned} \xi(k+1) &= A_\xi \xi(k) + B_\xi u(k) \\ y(k) &= C_\xi \xi(k) \end{aligned} \tag{3.10}$$

where:

$$A_\xi = \begin{bmatrix} A & B & \mathbf{0} & \dots & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \dots & \dots & \mathbf{0} \\ \vdots & \vdots & & \ddots & \dots & \vdots \\ \mathbf{0} & \dots & \dots & \dots & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \dots & \dots & \dots & \dots & \mathbf{0} \end{bmatrix}, B_\xi = \begin{bmatrix} \mathbf{0} \\ \vdots \\ \vdots \\ \mathbf{I} \end{bmatrix} \tag{3.11}$$

$$C_\xi = \begin{bmatrix} C & D & \mathbf{0} & \dots & \mathbf{0} \end{bmatrix}$$

Then classical MPC formulation of the tracking problem is:

$$\begin{aligned} \underset{\substack{(u(k),\dots,\\ u(k+N-1))}}{\text{minimize}} \quad & \sum_{i=1}^{N} \|y^{ref}(k+i) - y(k+i)\|_{q_y}^2 + \|\xi(k+i)\|_{Q_x}^2 + \|u(k+i)\|_R^2 \\ \text{subject to} \quad & \xi(k+i+1) = A_\xi \xi(k+i) + B_\xi u(k+i) \qquad \forall i \in \{0,\dots,N-1\}, \\ & y(k+i) = C_\xi \xi(k+i) \qquad \forall i \in \{1,\dots,N\}, \\ & \xi(k+i) \in \mathcal{X} \times \mathcal{U}^d \qquad \forall i \in \{1,\dots,N-1\}, \\ & u_l(k+i) \in \mathcal{U}_l \qquad \forall i \in \{0,\dots,N_L-1\}, \\ & \xi(k+N) \in \mathcal{X}_f \end{aligned} \tag{3.12}$$

where q_y, Q_x and R are weighting matrices, N is the prediction horizon, y^{ref} is the reference signal and $\mathcal{X}_f$ is the terminal positively invariant set and needs to be parameterized according to an admissible trajectory.

In the following, we propose a strategy based on the direct use of the Maximal Controllable Set (MCS) or its approximation. This set is defined as the largest set of initial states for which there exists an admissible sequence of control actions that makes the state trajectory to remaining in the set itself. This set will be denoted $\mathcal{C}$. In practice, this set can be approximated by the N-step Controllable Set C_N in polyhedral form. It can be construct from a positively invariant set through the procedure in [Nguyen et al., 2013].

3.2.2 Maximal Controllable Set based approach

The alleged control structure for time-delay linear systems applied to the driving simulation application preserves the MPC-based structure and is explained in the

following after the recall of the central definitions of controlled invariance and maximal controllable set that will be used in this section and the other chapters.

Definition 3.1: Controlled positive invariance

A set $\mathcal{C} \subset \mathcal{X}$ is said to be **controlled positively invariant** with respect to $x(k+1) = f(x(k), u(k))$ and the constraint $(\mathcal{X}, \mathcal{U})$ if for any initial state $x_0 \in \mathcal{C}$ there exists a control action $u(k) \in \mathcal{U}$ that makes the trajectory remaining in $\mathcal{C}$.

Definition 3.2: Maximal Controllable Set

The **maximal controllable set** denotes the largest controlled positive invariant set w.r.t a given controlled dynamics $x(k+1) = f(x(k), u(k))$ and sets of constraints $(\mathcal{X}, \mathcal{U})$.

Alleviated MPC

Considering the maximal controllable set $\mathcal{C}$, an alleviated formulation of the MPC problem is:

$$\begin{aligned}
\underset{\substack{(u(k),\dots,\\ u(k+N-1))}}{\text{minimize}} \quad & \sum_{i=1}^{N} \|y^{ref}(k+i) - y(k+i)\|^2_{Q_y} + \|x(k+i)\|^2_{Q_x} + \|u(k+i)\|^2_R \\
\text{subject to} \quad & x(k+i+1) = Ax(k+i) + Bu(k+i) \; \forall i \in \{0, \dots, N-1\}, \\
& y(k+i) = Cx(k+i) + Du(k+i) \qquad \forall i \in \{0, \dots, N\}, \\
& \mathbf{x(k+1)} \in \mathcal{C}, \\
& u(k+i) \in \mathcal{U} \qquad \forall i \in \{0, \dots, N-1\}
\end{aligned} \tag{3.13}$$

In other words, if we have a representation of the maximal controllable set, we can replace the condition of the predicted states belonging to $\mathcal{X}$ and the terminal condition by the condition on the one-step ahead state prediction to belong to the maximal controllable set.

Proposition 3.1

The MPC control based on 3.13 is recursively feasible independently of the realization of the reference signal.

Proof

This property is inherited from the linear structure of the prediction model for the delay-time system in the extended state space for which it is known that any domain of attraction for a linear constrained system is a tracking domain of attraction [Blanchini and Miani, 2000]. The constraint imposed in 3.13 is ensuring that the trajectories remain at each moment in time in a control invariant approximation of this domain. □

From the convergence point of view, theoretical properties are not enforced. Indeed, such an analysis would need a concept of best feasible reference with respect to which the convergence properties should be evaluated [Olaru and Dumur, 2005].

However, these convergence notions are less relevant in the MCA tracking as long as the tracking is not done with respect to fixed points and its high variability is making the performance index less reliable with respect to the motion sickness as this can be sensed by the internal ear and the otolithic subsystems.

Delay compensation and the corresponding MPC strategy

For the theoretical analysis of the control scheme, we consider the following generic LTI time-delay system with single output:

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k-d) \\ y(k) &= Cx(k) + Du(k-d) \end{aligned} \tag{3.14}$$

where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{1\times n}$ and $D \in \mathbb{R}^{1\times m}$.

The principle of this method is summarized in Figure 3.15:

- knowing the past inputs construct the free-response up to $x(k+d|k)$
- minimize the difference between the reference and the predicted output over the horizon $k+d, \ldots, k+N$
- impose as hard constraint $x(k+d+1|k) \in \mathcal{C}$.

The receding horizon optimization related to the MPC formulation is given below:

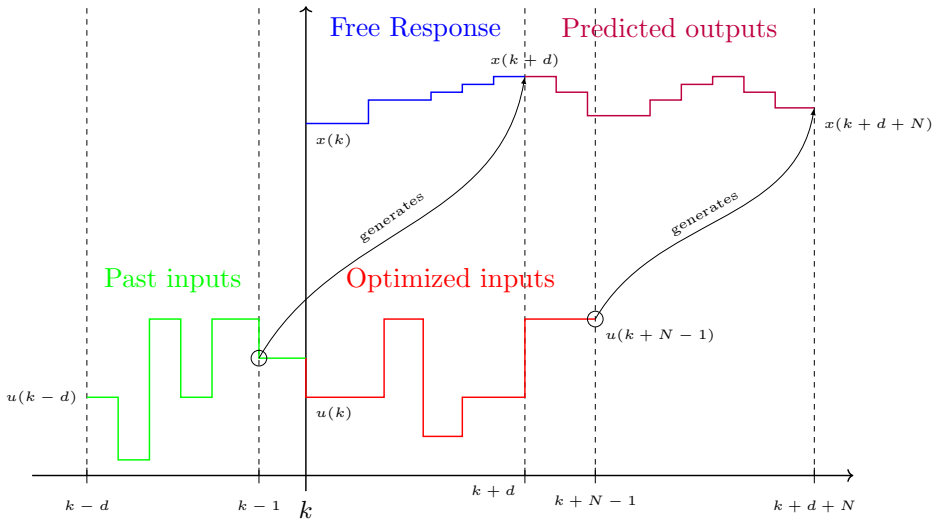

Figure 3.15: Principle of the strategy: the past inputs impose the free response up to $k+d$ while the optimized inputs are chosen such that the predicted inputs ensure the first predicted state (at stage $k+d+1$) to belong to the controllable set $\mathcal{C}$.

$$
\begin{aligned}
\underset{(u(k),\ldots,u(k+N-1))}{\text{minimize}} \quad & \sum_{i=d+1}^{d+N} \|y^{ref}(k+i) - y(k+i)\|_{q_y}^2 + \|x(k+i)\|_{Q_x}^2 + \sum_{i=1}^{N} \|u(k+i)\|_R^2 \\
\text{subject to} \quad & x(k+i+1) = Ax(k+i) + Bu(k+i-d), \\
& \qquad\qquad \forall i \in \{0,\ldots,d+N-1\}, \\
& y(k+i) = Cx(k+i) + Du(k+i-d), \\
& \qquad\qquad \forall i \in \{d+1,\ldots,d+N\}, \\
& x(k+d+1) \in \mathcal{C}, \\
& (u(k),\ldots,u(k+N-1)) \in \mathcal{U}
\end{aligned}
\tag{3.15}
$$

where q_y, Q_x and R are weighting matrices.

The first component of the solution is effectively applied. Those constraints guarantee the recursive feasibility of the controller while optimizing the cost function. The recursive feasibility of the controller can be guaranteed using the following results.

Lemma 3.1

Let $\mathcal{C}$ be an admissible controlled invariant set with respect to the delay-free dynamics: $\xi(k+1) = A\xi(k) + Bv(k)$, where (A, B) are the matrices of (3.14). If the current state $x(k) \in \mathcal{C}$ and the past inputs of (3.14) $\{u(k-d), \ldots, u(k-1)\} \in \mathcal{U}$ such that $\{x(k+1), \ldots, x(k+d)\} \in \mathcal{C}$, then there exists a control action $u(k)$ such that $x(k+d+1) = Ax(k+d) + Bu(k) \in \mathcal{C}$.

Proof

Let us start from the assumption that $x(k) \in \mathcal{C}$. By exploiting the existence of feasible control sequence such that $\{x(k+1), \ldots, x(k+d)\} \in \mathcal{C}$ one can concentrate on the delay-free dynamics:

$$\xi(k+1) = A\xi(k) + Bv(k)$$

which is equivalent, due to time-invariance, to

$$x(k+d+1) = Ax(k+d) + Bu(k)$$

For any $\xi(k) \in \mathcal{C}$ there exists $v(k) \in \mathcal{U}$ such that $\xi(k+1) \in \mathcal{C}$. By choosing $u(k) = v(k)$ one has $x(k+d+1) \in \mathcal{C}$ and the feasibility of the problem is ensured. □

Proposition 3.2

Given an initial state $x(0) \in \mathcal{C}$ such that $x(0) = \begin{bmatrix} p_0 & 0 & \theta_0 \end{bmatrix}^\intercal$ with $(p_0, \theta_0) \in \mathbb{R}^2$ and a past control sequence $\{u(-d), \ldots, u(-1)\} = \{0, \ldots, 0\}$, then the algorithm is recursively feasible.

Proof

Since the control actions are inactive at the system initialization, the trajectory is ensured to remain at the origin during d time steps. Moreover, since the origin belongs to the maximal controllable set, the Lemma 3.1 above can be applied to complete the proof.

With these elements, the practical use of the proposed approach in the driving simulators control framework is straightforward. Whenever the simulation session is initialized, the calibration is made to a position in the center of the environment

with inactive controls for a time window covering the considered delay in the dynamics. With such an initialization, the recursive feasibility is guaranteed for the nominal system.

Example: MPC delay

We consider the study case described in Section 2. The weight parameters use for the simulation are: $q_y = 10$, $Qx = diag(25, 1, 1)$, $R = diag(10, 1)$ and the prediction horizon: $N = 100$. The acceleration rendering is compared to the reference profile on Fig.3.16 by means of its two components. We observe the acceleration globally follows the profile with a lower amplitude by noticing that a part of acceleration is proportional to the tilt angle. An important feature of the acceleration rendering is the restitution of the shape of the reference profile, particularly during the slalom phase (there is no saturation due to inputs constraints), then the driver feels the correct variations. Moreover the delay has been compensated which can cancel the motion sickness. The Figure.3.17 depicts the displacement of the lateral rail, its speed and the tilt angle profile as well as their physical limitations that are actually satisfied along the simulation. The acceleration of the rail (represented on Fig.3.18) is added to the tilt angle contribution. We can observe that the rail contributes on the fast varying components of the slalom phase while the tilt impacts more the slow turn phase (which can be explained by the constraints on tilt rate).

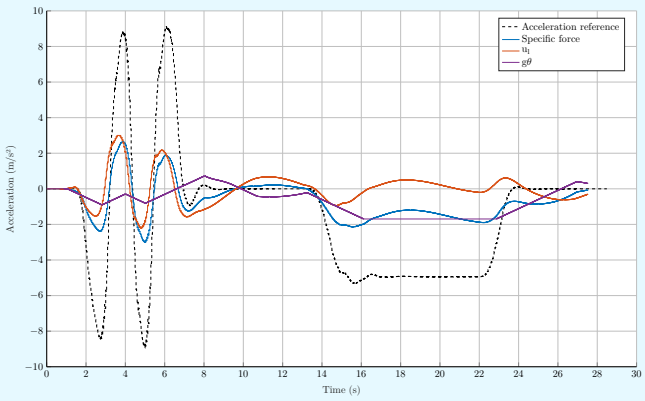

Figure 3.16: Specific force sensed by the driver

Limitations of the rail are underlined on Fig.3.17 and on Fig.3.19 where the rail part of the state space is represented (position and speed) with the trajectory within Maximal Controllable Set, which shows that the closed-loop dynamics are conservative in the presence of time delay. Fig.3.20 depicts the computation time at each iteration.

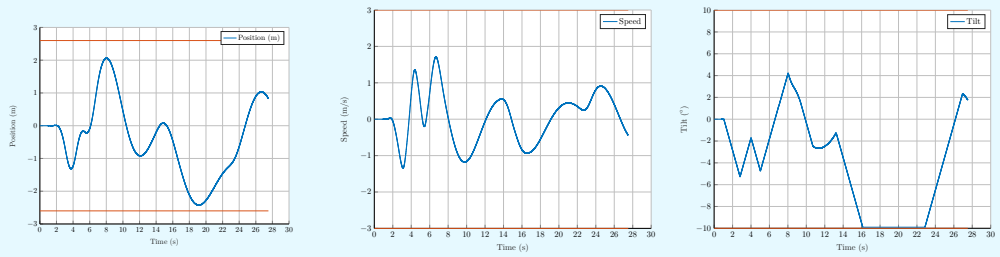

Figure 3.17: Position of rails (left), Speed of rail (center), tilt angle (right) in function of time

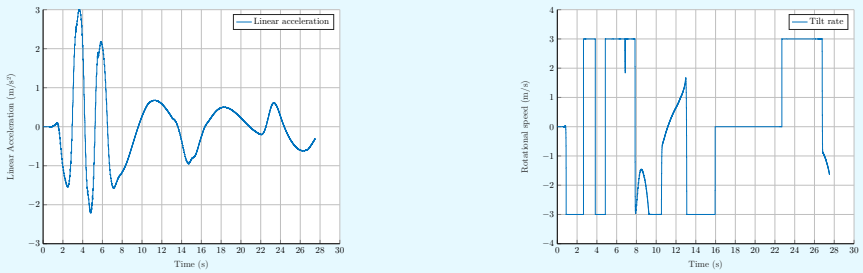

Figure 3.18: Inputs in function of time: Linear acceleration (right) and tilt rate (left)

The Figure.3.20 represents the computation time at each time instant.

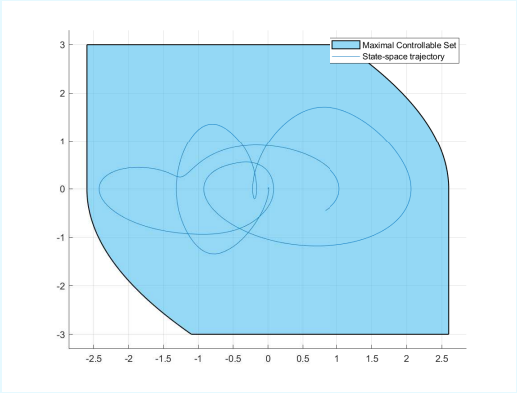

Figure 3.19: Projection of the state space trajectory on the axis position-velocity

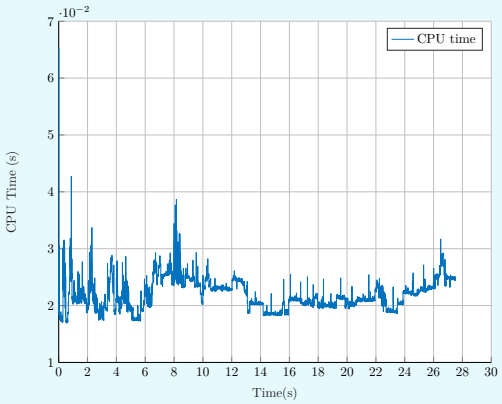

Figure 3.20: Computation time

3.2.3 Discussion and conclusion

Discussion

First, aside the practical advantages of the delay compensation, it should be said that the proposed strategy is limited by the complexity of the approximation of the maximal controllable set. Indeed, the complexity of the optimization problem (3.15) depends on the complexity of $\mathcal{C}$. The curse of dimensionality can be expected as the dimension of the state-space increases and thus the whole complexity of the procedure making the real-time performance unreachable. This remark triggers the developments in the next sections in the quest for a replacement for this complex ingredient (as well ass the terminal set) in the MPC design

In our application the inclusion of the $(d+1)th$ state within $\mathcal{C}$ may imply that the controller forces punctually the trajectory to remain in $\mathcal{C}$ by applying the maximal input value. This can be illustrated on a slightly different scenario of Simulation with respect to the one presented in the previous section. Consider a less conservative design with a lower weight on the position component of the cost function: $Qx = diag(\begin{bmatrix}15 & 1 & 1\end{bmatrix})$. We can check on the Figure.3.22 that the trajectory uses a larger area and benefits from the whole workspace to improve the acceleration rendering depicted on Figure.3.21. However, after 20s one can observe an important profile increase in the rail acceleration to maintain the simulator position within the bounds that impacts the overall response. If this behaviour is short enough, it may be filtered by the inner ear and the impact on the driver is mitigated. Obviously this depends on the sensibility characteristic of the driver physiology. As a first conclusion there is a trade-off between the conservativeness of the design (weights on states and inputs) and the implication of delayed states

constraints.

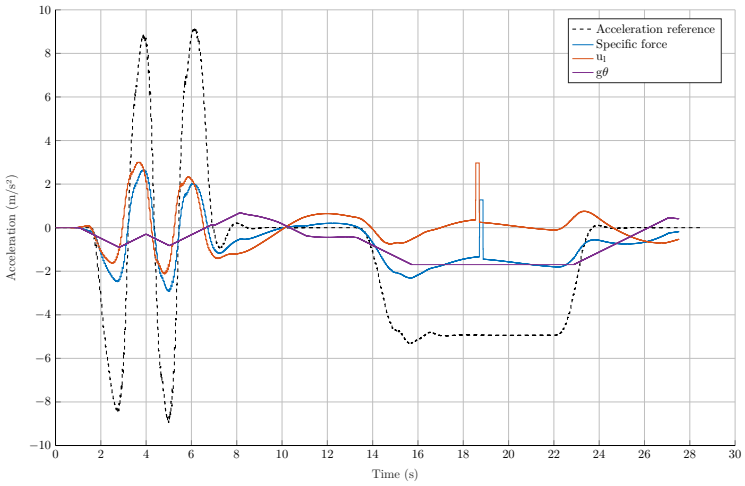

Figure 3.21: Specific force sensed by the driver

Conclusion

Time delays are inherent to the structure of driving simulators due to the mechanical inertia and the communication protocol between the algorithms which render the virtual scene and the physical move of the platform, all of them having a joint action on the human senses. The motion sickness being directly related to the time-delay, its inclusion in the control algorithm needs to be optimized in order not to further deteriorate the human perception.

The classical optimal control approach based on extended state space (3.12) is difficult to implement on real-time systems because of the complexity inherited from the increased dimension of parameters. In this section we proposed an alleged control strategy based on the knowledge of the maximal controllable set (3.15) by enforcing the delayed states to remain within this safe set at a reduced computational cost.

This strategy is recursively feasible and globally follows the expected acceleration while satisfying states and inputs constraints. However its performance can be limited on one side by the size (topology) of the controllable set and on the other side by the policy of constraints activation among the control channels. This latter drawback can be dealt with, by the choice of weights in the design of the strategy. In the next sections we aim to move the constrained control design from

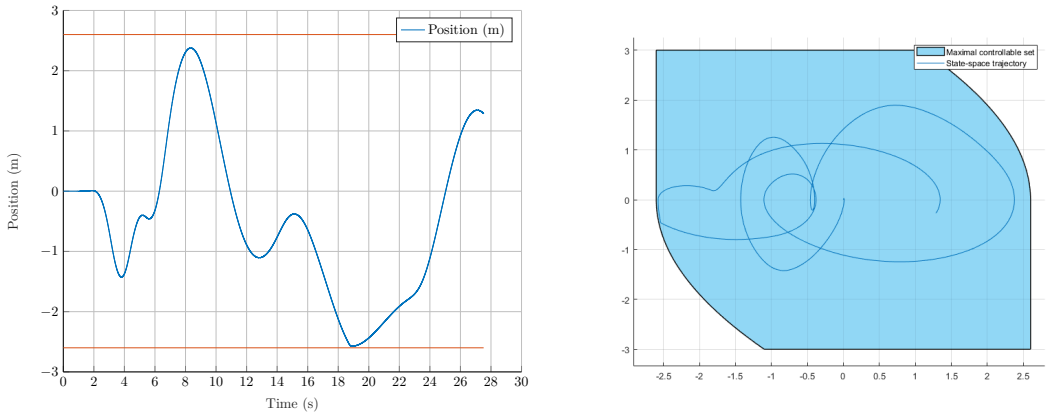

Figure 3.22: Left: position of the platform. Right: State space trajectory

the MPC framework in order to explore new formulations able to preserve the main feasibility domain all by providing low complexity alternatives for tracking.

Chapter 4

Interpolation based control for tracking

Contents

In the field of constrained control, merging control Lyapunov functions [Grammatico et al., 2013], sharing the control authority between several feedback laws [Bacic et al., 2003, Grammatico et al., 2013] or interpolating among control actions [Nguyen, 2014, Kheawhom and Bumroongsri, 2014] are based on the same concepts and received the interest of the control community as a versatile (optimization-based) design technique to avoid the classical Model Predictive Control formulations all by preserving the essential features: recursive feasibility, stability, continuity and manageable performance certificates for the unconstrained control regions.

Originally, *interpolation-based control* (IBC) has been established [Nguyen et al., 2013] to enhance Vertex Control method [Gutman and Cwikel, 1986] which was capable to stabilize a constrained linear system exclusively based on the knowledge of the feasible control actions on the boundary of a controlled invariant set. For obvious reasons, this controlled invariant set was used to approximate the maximal controlled invariant set.

The main novelty behind the IBC methodology [Nguyen et al., 2011a] is to consider aside the controlled invariant set which ensures a large domain of attraction, one (or multiple) controlled invariant sets, associated to high feedback gains (associated with high performance in the unconstrained case) in the control design. The convex decomposition between a high gain unconstrained linear feedback law in the neighborhood of the equilibrium and a low gain Vertex control law on the boundary of the domain of attraction was shown to preserve the computationally-attractive LP formulation for LTI systems and add the performance on top of the large feasible domain. All these developments reached a maturity for the constrained regulation around the origin and have been applied in different domains [Ballesteros-Tolosana et al., 2016, Tuchner and Haddad, 2017] with different groups developing the associated design tools [Scialanga and Ampountolas, 2019]. Given all these interesting properties, the IBC is a natural candidate for the MCA design. Interesting though, the trajectory tracking was not addressed in the previous works and represents an important aspect to complete the technique with the capabilities of a generic constrained control routine. The goal of the present chapter is to bridge this gap and contribute to the theoretical foundation of the IBC on one side and to apply the resulting technique on the MCA design.

Before entering in the IBC tracking details, let us recall that constrained trajectory tracking is a topic which received the interest over more than 30 years [Bemporad, 1998, Garone et al., 2017], often in connection with Model Predictive Control [Limon and Alamo, 2013, Falugi, 2015]. The underlying problem in the constraint trajectory tracking is the description of the feasible range of trajectories given a predefined stabilizing control able to handle the constraints and guarantee the stabilization [Olaru and Dumur, 2005, Limon et al., 2005]. Once this objective is achieved, the selection among the feasible trajectories of a suitable candidate is usually made with respect to an optimization-based procedure and carries different names as for example reference-governor[Bemporad, 1998], reference-management or virtual trajectory selection[Santos, 2018, Limón et al., 2008]. Once the recursive feasibility is ensured, the techniques vary with respect to the strategies employed and depend on the anticipative information on the reference to be tracked. If this reference takes the form of set-points that are piecewise constant, then the convergence properties can be sought. If the reference trajectory is varying fast or its description is not receding along the system functioning, then the tracking

performance can be addressed on probability, in economic terms or can be seen as measures of robustness.

Coming back to the core of the present chapter, IBC trajectory tracking problem will be formulated and solved starting from the classical ingredients of a IBC regulation. The design principle is inherited from reference governor mechanisms which design an admissible reference with respect to the static constraints and for which the regulation capabilities of the IBC can be fully exploited. For the class of linear time-invariant systems, the generation of an admissible virtual trajectory is done in conjunction with a scaling mechanism for the controlled invariant sets involved in the IBC design. Globally the tracking IBC solution is shown to preserve the LP structure and thus presents attractive features for real-time implementation. The proposed technique is formally presented together with the feasibility proof and a series of enhancements are presented along with numerical examples.

4.1 Interpolation Based Control

4.1.1 Framework

Given the constrained linear discrete-time system :

$$x(k+1) = Ax(k) + Bu(k) \tag{4.1}$$

where $x \in \mathcal{X} \subset \mathbb{R}^n, u \in \mathcal{U} \subset \mathbb{R}^m$, $A \in \mathbb{R}^{n\times n}, B \in \mathbb{R}^{n\times m}$. $\mathcal{X}$ and $\mathcal{U}$ are sets of constraints respectively on states and control vector both containing the origin in their interior.

Assumption 4.1: Inner and outer sets

We assume that the system (4.1) is controllable. We also consider two controllable sets $\Omega^o \subset \mathcal{X}$ and $\Omega^v \subset \mathcal{X}$ which are convex compact sets and satisfy:

$$\Omega^o \subset \Omega^v \subset \mathcal{X} \tag{4.2}$$

Ω^o will be denoted *inner set* and Ω^v the *outer set*. Finally, we assume that each of them has its own associated constrained control law:

$$u^o = \mathcal{K}^o(x) \text{ if } x \in \Omega^o \text{ and } u^v = \mathcal{K}^v(x) \text{ if } x \in \Omega^v/\Omega^o \tag{4.3}$$

such that $\mathcal{K}^o(\Omega^o) \subset \mathcal{U}$ and $\mathcal{K}^v(\Omega^v) \subset \mathcal{U}$.

Remark. *The* inner set *is generally associated with a high gain control law while the* outer set *is intended to approximate the maximal controllable set within* $\mathcal{X}$. *So*

for every initial state $x_0 \in \Omega^v$ there exists a control sequence that leads the system's trajectory to the origin and consequently to Ω^o in a finite number of steps.

4.1.2 Principle

Summary

At each time step $k \in \mathbb{N}$, given the measurement of the current state, there exists a convex combination of states $x^o(k) \in \Omega^o$ in the inner invariant set and respectively $x^v(k) \in \Omega^v$ in the outer controlled invariant set such that :

$$x(k) = c(k)x^v(k) + (1 - c(k))x^o(k) \tag{4.4}$$

where $c(k) \in [0, 1]$ is a convex factor. Thus, a control action can be devised, also using a convex combination of the control laws (4.3):

$$u(k) = c(k)u^v(k) + (1 - c(k))u^o(k) \tag{4.5}$$

The convex factor $c(k)$ is chosen to maximize the contribution of the local controller $\mathcal{K}^o(.)$. Consequently $c(k)$ and the components $x^o(k)$ and $x^v(k)$ can be computed as the solution of the nonlinear optimization problem:

$$\begin{aligned} \begin{bmatrix} x^v & x^o & c^* \end{bmatrix}^T = \underset{(x^v,x^o,c)(k)}{\text{minimize}} \quad & c \\ \text{subject to} \quad & x^v \in \Omega^v, x^o \in \Omega^o, \\ & x(k) = cx^v + (1 - c)x^o, \\ & c \in [0, 1]. \end{aligned} \tag{4.6}$$

The procedure of IBC is explicited in Algorithm 1

The block diagram on Figure 4.1 summarizes the procedure where the grey part corresponds to the IBC block that will be used later. An example of the geometrical properties in $\mathbb{R}^2$ is given on Figure 4.2 where the shapes of the sets are similar to what can be computed for a discrete-time polyhedral constrained double integrator. The red set Ω^o would be practically designed as a positively invariant set included in the Maximal Controllable Set (MCS) which plays the role of Ω^v in blue. The figure also depicts the geometrical interpretation of the decomposition (4.4) of the state $x(k) \in \Omega^v/\Omega^o$. There exists a priori a infinite number of couples $(x^v(k), x^o(k))$ satisfying(4.4) but as we seak to minimize the convex factor $c(k)$, the optimal configuration $x^v(k)$ is on the frontier of Ω^v and $x^o(k)$ on the frontier of Ω^o which enhance the performance of the constrained control as explained in Theorem 4.1.

Algorithm 1: Interpolation Based Control

Input : Ω^o,Ω^v,$\mathcal{K}^o$,$\mathcal{K}^v$,$x_0 \in \Omega^v$, N_{simu},(A,B),$\mathcal{X}$,$\mathcal{U}$
Output: $(x(k))_{k=0,\dots,N_{simu}}$,$(u(k))_{k=0,\dots,N_{simu}-1}$

$k = 1$
while $k < N_{simu}$ **do**
  Solve (4.6) $\longrightarrow c^*, x^{v*}, x^{o*}$
  $c(k) \leftarrow c^*$, $x^v(k) \leftarrow x^{v*}$, $x^o(k) \leftarrow x^{o,*}$
  Compute $u^v(k) = \mathcal{K}^v(x^v(k))$, $u^o(k) = \mathcal{K}^o(x^o(k))$
  Compute $u(k) = c(k)u^v(k) + (1 - c(k))u^o(k)$
  $x(k+1) = Ax(k) + Bu(k)$
  $k \leftarrow k + 1$
end

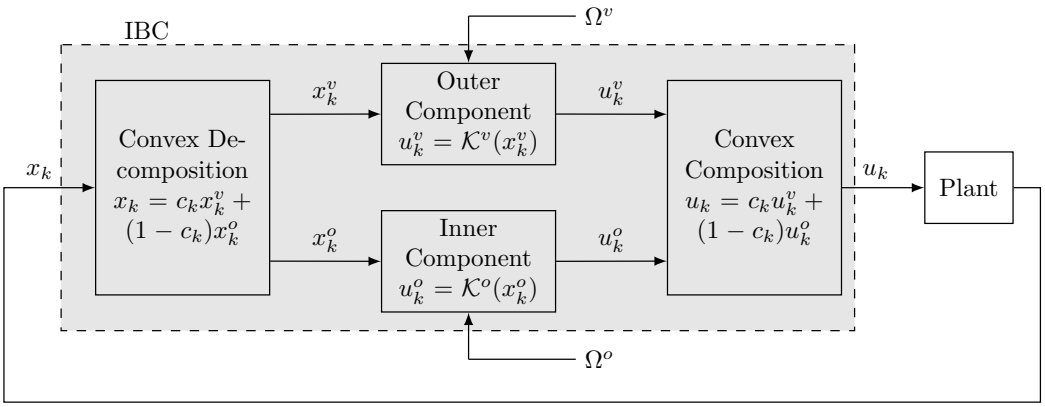

Figure 4.1: Block diagram of IBC Principle

Theorem 4.1: Decomposition on the frontiers [Nguyen et al., 2011b]

The optimal solution $(x^v(k), x^o(k), c(k))$ of the optimization problem (4.6) is such that $x^v(k)$ and $x^o(k)$ are respectively on the frontier of Ω^v and Ω^o.

Proof

Let $x \in \Omega^v/\Omega^o$ and $(x^v, x^o, c) \in \Omega^v \times \Omega^o \times]0,1[$ such that $x = cx^v + (1-c)x^o$.

- If x^o is strictly included in Ω^o, we consider $\tilde{x}^o$ as the intersection of Ω^o

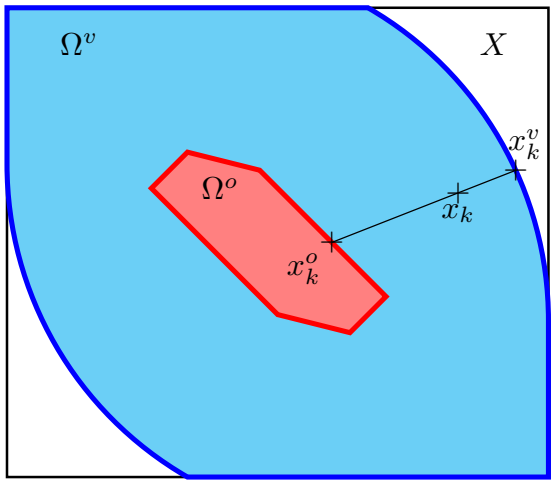

Figure 4.2: Geometric interpretation of IBC in the state space

with the segment linking x and x^o.

$$\tilde{x}^o = Fr(\Omega^o) \cap [x^o, x]$$

As $\tilde{x}^o \in [x^o, x]$, there exists $\tilde{c}$ such that $x = \tilde{c}\tilde{x}^v + (1 - \tilde{c})x^o$. Since Ω^o is convex, we have necessarily $\tilde{c} < c$.

- If x^v is strictly included in Ω^v, we consider $\hat{x}^v$ as the intersection of Ω^v with the segment linking x and x^v.

$$\tilde{x}^v = Fr(\Omega^v) \cap [x^v, x]$$

As $\hat{x}^v \in [x^v, x]$, there exists $\hat{c}$ such that $x = \hat{c}x^v + (1 - \hat{c})\hat{x}^o$. Since Ω^o is convex, we have necessarily $\hat{c} < c$. □

In both cases the convex factor is minimized when x^o and x^v are located on the frontier of their sets. This property can be used in a more general framework with multiple interpolation factors, corresponding to tuples of states on the frontier of their respective controlled invariant sets.

With these elements, the construction of an optimization-based control procedure for constrained-control becomes apparent but aside the relative low complexity of the sets involved, the stability and the computational advantages need further analysis as shown next.

4.1.3 Properties: Feasibility and stability

Recursive Feasibility

Theorem 4.2: Recursive feasibility of IBC

Given the system (4.1) and the controlled invariant sets Ω^o and Ω^v with $\Omega^o \subset \Omega^v$, the following control law :

$$u(k) = c^*(k)u^v(k) + (1 - c^*(k))u^o(k) \tag{4.7}$$

where $u(k)^o$ is a control action in Ω^o and $u^v(k)$ is a control law in Ω^v and the convex factor $c^*(k)$ is the solution of the optimization problem (4.6) at time k is admissible and the origin is a stable equilibrium for the closed-loop system with a basin of attraction Ω^v.

Proof

First, thanks to the constrained control laws in Ω^o and Ω^v, we have:

$$u(k) = c^*(k)\underbrace{u^v(k)}_{\in\mathcal{U}} + (1 - c^*(k))\underbrace{u^o(k)}_{\in\mathcal{U}}$$

It results that the IBC control law $u(k)$ is admissible ($u(k) \in \mathcal{U}$). Second, we prove that $x(k+1)$ remains within Ω^v.

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ &= A(c^*(k)x^v(k) + (1 - c^*(k))x^o(k)) \\ &\qquad + B(c^*(k)u^v(k) + (1 - c^*(k))u^o(k)) \\ &= c^*(k)\underbrace{(Ax^v(k) + Bu^v(k))}_{\in\Omega^v} + (1 - c^*(k))\underbrace{(Ax^o(k) + Bu^o(k))}_{\in\Omega^o\subset\Omega^v} \end{aligned}$$

□

Stability

Theorem 4.3

The IBC controller given by (4.7) is stable if Ω^v and Ω^o are controlled contractive with respect to their associated controller ($i.e\ \forall x \in \Omega^v\ \exists u^v \in \mathcal{U}$ s.t $u^v = \mathcal{K}^v(x)$ and $Ax + Bu^v \in \lambda\Omega^v, \lambda \in [0, 1[$)

Proof

The optimized interpolation factor c^* is a Lyapunov function in Ω^v/Ω^o. Furthermore, there is a local Lyapunov function in Ω^o which is controlled invariant and thus the trajectories entering in this set asymtotically converge to the origin. The formal proof can be found in [Nguyen et al., 2013] □

The previous Theorem builds on the contractive assumptions on the controlled invariant set Ω^v which can be further relaxed to the existence of a finite index N such that the $N-$step forward trajectories initiated on the boundary of Ω^v reach the interior of Ω^v. We do not dwell here on these theoretical notions but underline the fact that the assumptions needed for the implementation of a IBC are mild.

Once these principles and main stability properties are established, one can move toward the practical implementation and real-time application by providing an effective construction of the control actions on the boundary of the outer controlled invariant set.

In [Nguyen et al., 2013], the authors demonstrate the stability of the closed-loop using Algorithm 1 in a specific case where the following assumptions hold

- the constraints $\mathcal{X}, \mathcal{U}$ are polyhedral
- the inner set Ω^o is positively invariant with respect to a linear feedback.
- the outer controller is a *Vertex controller* whose operating is given below

Definition 4.1: Vertex Control [Gutman and Cwikel, 1986]

Given a controlled invariant set Ω^v and the dynamical system (4.1), for any state $x \in \Omega^v$ a constrained control action driving x toward the origin can be computed by the following optimization problem:

$$\begin{aligned} \underset{\lambda,u}{\text{minimize}} \quad & \lambda \\ \text{subject to} \quad & Ax + Bu \in \lambda\Omega^v, \\ & u \in \mathbb{U}, \\ & \lambda \in [0,1] \end{aligned} \tag{4.8}$$

In other words, the vertex controller seeks a control action that pull the trajectory away from the frontiers of Ω^v. The vertex control law guarantees recursive feasibility and asymptotic stability (by using λ^* solution of (4.8) as a Lyapunov function).

Thus, the stability property is guaranteed and proved in Theorem 4.4.

Theorem 4.4: Stability of IBC

Given the previous assumptions, the IBC procedure is asymptotically stable.

Proof

We first consider the Lyapunov candidate V such that:

$$V(x(k)) = c^*(k)\ \forall x(k) \in \Omega^v/\Omega^o$$

where $c^*(k)$ is the solution of (4.6) at instant k. We also have:

$$x(k+1) = c^*(k)(Ax^v(k) + Bu^v(k)) + (1 - c^*(k))(Ax^o(k) + Bu^o(k))$$

by setting $x^v(k+1) = Ax^v(k) + Bu^v(k)$, $x^o(k+1) = Ax^o(k) + Bu^o(k)$ and $c(k+1) = c^*(k)$ we obtain the admissible decomposition:

$$x(k+1) = c(k+1)x^v(k+1) + (1 - c(k+1))x^o(k+1)$$

Since $[x^v(k+1)\ x^o(k+1)\ c(k+1)]^T$ is a feasible solution for the convex decomposition, the solution $[x^{v*}(k+1)\ x^{o*}(k+1)\ c^*(k+1)]^T$ of (4.6) verifies $c^*(k+1) \leq c^*(k)$. Then V is decreasing but not strictly, however we can compare this candidate to the Lyapunov function of the vertex controller (4.8). By noticing that the vertex controller is a particular case of the interpolation based controller by choosing the feasible convex factor $c(k) = 1\ \forall k$. Since the vertex controller is asymptotically stable, it leads the system to Ω^o in finite time. Finally, as Ω^o is a positive invariant set, the local controller which is also a particular case of IBC ($c(k) = 0 \forall k$) stabilizes the trajectory around the origin.

4.1.4 LP based implementation of IBC

We now address the implementation of IBC procedure in view of our application by assuming that Ω^v and Ω^o are polyhedral sets defined by:

$$\Omega^v = \{x \in \mathbb{R}^n / \ F_v x \leq g_v\} \text{ and } \Omega^o = \{x \in \mathbb{R}^n / \ F_o x \leq g_o\} \tag{4.9}$$

In this case, the bilinear problem (4.6) becomes a LP problem (see Definition A.16) thanks to the variable change:

$$r^v = cx^v \tag{4.10}$$

Thus, the convex decomposition is done by the following LP problem:

$$\begin{bmatrix} r^{v*} & c^* \end{bmatrix}^T = \underset{(r^v,c)(k)}{\arg\min} \quad c \\ \text{subject to} \quad F_v r^v \leq c g_v, \\ F_o(x(k) - r^v) \leq (1-c) g_o, \\ c \in [0,1] \tag{4.11}$$

Finally, one has to distinguish three cases:

- $c^* \in]0,1[$, then $x^{v*} = r^{v*}/c^*$ and $x^{o*} = (x - r^{v*})/(1-c^*)$
- $c^* = 0$, consequently, $x(k) \in \Omega^o$, $x^{v*} = 0$ and $x^{o*} = x(k)$
- $c^* = 1$, so $x(k)$ is on the outer boundary of Ω^v, $x^{o*} = 0$ and $x^{v*} = x(k)$

Consequently the Algorithm 1 can be implemented for real-time applications by leveraging the maturity of the LP solvers.

Example: IBC

Given the following discrete-time double integrator:

$$x(k+1) = \begin{bmatrix} 1 & 0.008 \\ 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} 0.000032 \\ 0.008 \end{bmatrix} u(k) \tag{4.12}$$

We chose as an inner set Ω^o the maximal positively invariant associated to the LQ controller:

$$u^o(x) = -\begin{bmatrix} 21.70 & 7.27 \end{bmatrix} x \tag{4.13}$$

as explained in Definition A.22. The outer set Ω^v is the N-step controllable set leading to Ω^o in $N_s = 184$ steps and associated to a Vertex controller (4.8). The two sets are depicted on Figure 4.3.

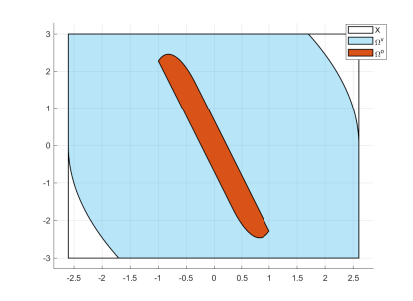

Figure 4.3: Representation of Ω^o and Ω^v in the state-space

We aim to compare here IBC with classical MPC controller such as (2.18) with a large prediction horizon $N = 200$ and using Ω^o as a terminal set. The trajectories of the two

controllers are depicted on Figure 4.4 while the control signals and the computation times are represented on Figure 4.5.

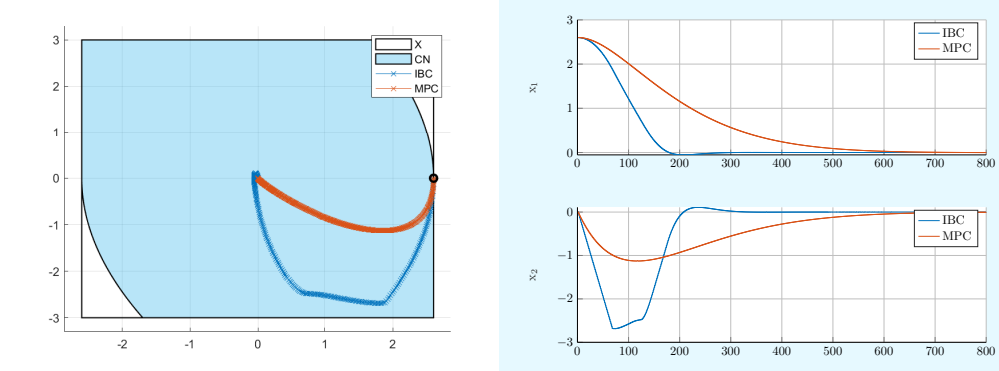

Figure 4.4: Left: State space trajectories. Right: trajectories in the time domain

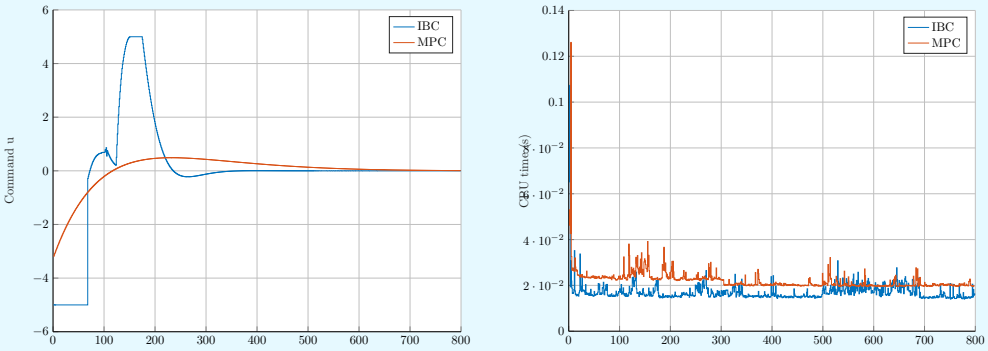

Figure 4.5: Left: Control signals. Right: Execution time

One can notice the higher speed of convergence of the trajectory toward the origin in the IBC case with a lower computationnal burden.

4.2 Interpolation Based Control for tracking

4.2.1 Framework/ Command governors

The increase of computational capacities allowed the use of MPC in order to handle constraints. When this is used in a tracking problem, it aims to minimize a tracking error over a time horizon using the prediction of future states based on system model (2.18). The last constraint inherited from the classical regulation problems

imposes the predicted state to reach the positively invariant subset $\mathcal{X}_f \subset \mathbb{R}^n$ to guarantee the recursive feasibility of the whole procedure. However, in the tracking framework such a constraint is drastically reducing the performances and can even jeopardize the feasibility in itself if it has to be considered jointly with a constrained on the tracking error. In order to cope with these shortcoming, the terminal constraints are generally parameterized and replaced with an invariant set adjusted in accordance with the reference to be tracked.

If the reference has a known dynamics that can be exploited in the prediction model, but the inclusion of reference dynamics makes the prediction and the optimization more difficult in term of computational burden as long as those reference signals are considered as part of the parameter vector. The reference signal can also imply a response that do not satisfy the constraints, the reference and command governors modify the reference so that no infeasibility occurs. The scheme on Fig. 4.6 summarizes constraint tracking operating, the plain part concerns the case where the reference is feasible, the controller works normally whereas the dashed part concerns the Reference Governor (RG) that modifies the reference. One simple solution consists in reducing the reference, thus the controller aims to reproduce a proportion of the reference signal given by the optimization problem:

$$\begin{aligned} \underset{\lambda}{\text{maximize}} \quad & \lambda \\ \text{subject to} \quad & \text{Feasibility of (2.22) for } r = \lambda y^{ref} \text{ as reference,} \\ & \lambda \in [0, 1] \end{aligned} \tag{4.14}$$

A second approach consists in finding the closest admissible reference thanks to the resolution of an optimization problem:

$$\begin{aligned} \underset{r}{\text{minimize}} \quad & \|y^{ref} - r\|_{Q_y}^2 \\ \text{subject to} \quad & \text{Feasibility of (2.22) for } r \text{ as reference} \end{aligned} \tag{4.15}$$

In the regulation case, Interpolation Based Control offers similar performance as MPC with a least computational burden, consequently the idea of this work is to develop a tracking procedure using IBC philosophy.

To introduce the IBC-based tracking procedure, we recall a theorem on homogeneity of controllable invariant sets that will be used further. The main results are then presented first by principles and subsequently through the mathematical formulations.

4.2.2 Preliminaries

Before stating the main constructive results towards a IBC strategy for tracking, let us recall some basic properties of the controlled invariant sets.

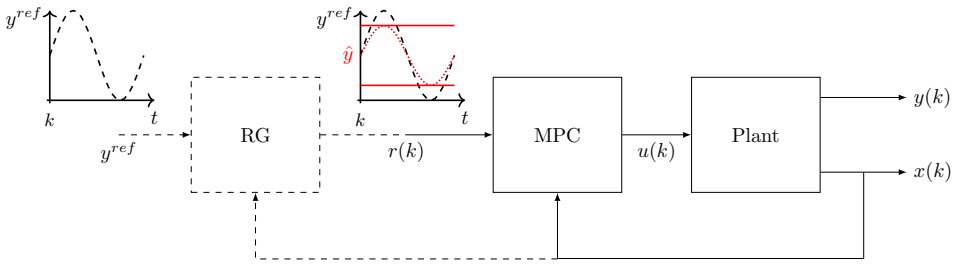

Figure 4.6: Constrained tracking using MPC

Lemma 4.1: Homogeneity of Controlled Invariance

If a given set Ω is controlled invariant with respect to (4.1), $\mathcal{U}$ and $\mathcal{X}$ as input and state constraints sets, then for all $\alpha \in [0,1]$ $\alpha\Omega$ is a controlled invariant set with an admissible control action in $\alpha\mathcal{U}$.

Proof

Let Ω be a controlled invariant set and $\alpha \in [0,1]$. If $\alpha = 0$, then $\alpha\Omega = \{0\}$ and the trivial choice $u = 0$ renders the set $\alpha\Omega$ controlled invariant. For $\alpha \in]0,1]$ and $\xi_0 \in \alpha\Omega$, there exists $x_0 \in \Omega$ such that $x_0 = \xi_0/\alpha$. Due to the controlled invariance of Ω, there exists $u_0 \in \mathcal{U}$ such that :

$$\begin{aligned} x_1 &= Ax_0 + Bu_0 \in \Omega \\ \xi_1/\alpha &= A(\xi_0/\alpha + Bu_0) \in \Omega \\ \xi_1 &= A\xi_0 + B\alpha u_0 \in \alpha\Omega \end{aligned}$$

By writing $v_0 = \alpha u_0$:

$$\text{There exists } v_0 \in \alpha\mathcal{U} \text{ such that } \xi_1 = A\xi_0 + Bv_0 \in \alpha\Omega$$

4.2.3 Principle

The challenge in the IBC case which is based on local controlled invariant sets, is their translation along the trajectory to be tracked which may lead to the loss of control invariance in the presence of input and state constraints.

In the following procedure, the translation of the sets will be accompanied by a re-scaling of those sets thus enforcing the feasibility properties.

The Interpolation Based Tracking (IBT) strategy will construct a control action

$u(k)$ to follow the reference as a trade-off between a control action $\tilde{u}(k)$ that generates a new virtual feasible reference trajectory for the system and a control action $v(k)$ that compensates the tracking error between the new reference and the current state with an IBC regulation with

$$u(k) = \tilde{u}(k) + v(k).$$

The procedure is illustrated in the Fig4.7.

Stage 1: *Reference Governor*

- At step k, the reference governor finds $\tilde{u}(k)$ and a scaling factor $\alpha(k)$ such that the virtual state $\tilde{x}(k+1|k)$ minimizes the distance to the real reference at the next step.
- The virtual state will satisfy the same dynamics as the nominal system at least for the current state.
- The virtual state is constrained to a neighborhood of the current state such that the error
$$\varepsilon(k) = x(k) - \tilde{x}(k) \tag{4.16}$$
is located in the re-scaled set $\alpha(k)\Omega^v$ which is a controlled invariant according to Lemma 4.1.
- This re-scaled controlled invariant centered on the virtual state has to be included in the global controlled invariant set Ω^v. This will be one of the main ingredients to prove the satisfaction of the global constraints.
- The contribution of the reference governor is higher if the current state $x(k)$ and the virtual state $\tilde{x}_{k|k}$ are close i.e if the scaling factor $\alpha(k)$ is small.
- This action, $\tilde{u}(k)$, has to lead the virtual state in another feasible neighborhood.

Stage 2: *Convex decomposition*
The IBC procedure is applied to the error dynamics:

$$\varepsilon(k) = x(k) - \tilde{x}(k)$$

in the re-scaled sets $\alpha\Omega^o$ and $\alpha\Omega^v$. A regulation action $v(k)$ is found based on standard IBC.

Stage 3: *Convex combination*
The applied control action $u(k)$ is the combination of the reference governor control vector $\tilde{u}(k)$ and the regulation action $v(k)$.

The block diagram on Figure 4.7 summarizes the procedure. One can notice that the global structure aims to drive the IBC block given on Figure 4.1 (filled in grey) thanks to the outputs of the reference governors.

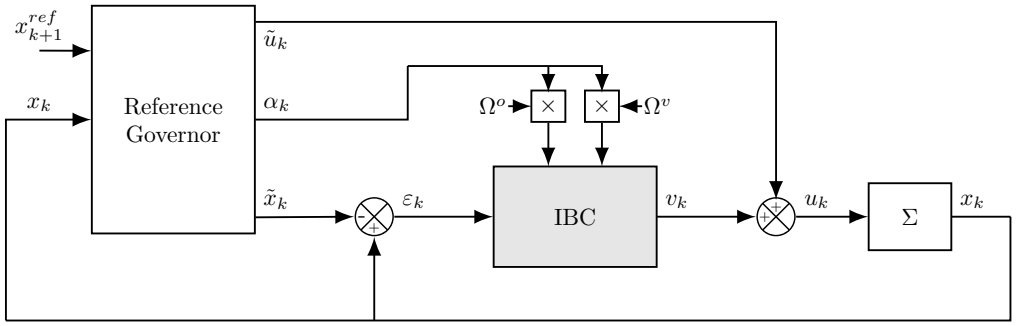

Figure 4.7: Block diagram of IBC for tracking

The Figure 4.8 provides a geometrical interpretation of the reference governor role based on the illustrative scheme of IBC of the Figure 4.2. At the instant k, the reference governor builds the set $\tilde{x}(k) \oplus \alpha(k)\Omega^v$ including the current state $x(k)$ such that the one step ahead set $\tilde{x}(k+1) \oplus \alpha(k)\Omega^v$ using the current scaling factor $\alpha(k)$ remain within the boundaries of Ω^v.

In other words, the reference governor finds and builds an admissible neighborhood for the current state where the IBC procedure can be done on the tracking error (4.16).

4.2.4 Mathematical formulation of the approach

Step 1: Reference Governor

Assume the current state $x(k)$ to be in the controlled invariant Ω^v, the reference governor if implemented through the resolution of the following optimization problem at each sampling time:

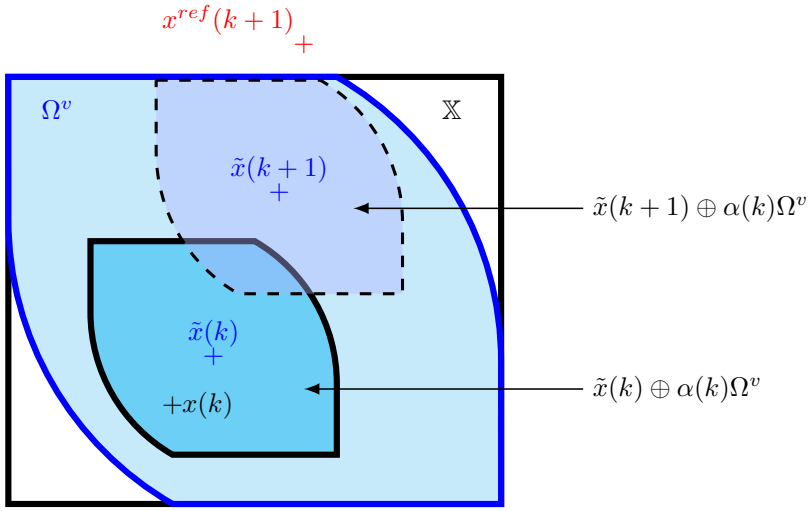

Figure 4.8: IBC for Tracking Geometric interpretation

$$\begin{aligned}
\begin{bmatrix}\tilde{u}(k)\\ \alpha(k)\end{bmatrix} = \underset{(\tilde{u}(k),\alpha(k))}{\arg\min} \quad & \|x^{ref}(k+1) - \tilde{x}(k+1|k)\|_Q^2 \\
\text{subject to} \quad & \tilde{x}(k+1|k) = A\tilde{x}(k|k) + B\tilde{u}(k), \\
& x(k) \in \{\tilde{x}(k|k)\} \oplus \alpha(k)\Omega^v, \\
& \{\tilde{x}(k|k)\} \oplus \alpha(k)\Omega^v \subset \Omega^v, \\
& \tilde{u}(k) \in (1-\alpha(k))\mathcal{U}, \\
& \{\tilde{x}(k+1|k)\} \oplus \alpha(k)\Omega^v \subset \Omega^v
\end{aligned} \tag{4.17}$$

The results of the optimization (4.17) lead to a solution $\tilde{u}(k)$ which provides practically a virtual trajectory $\tilde{x}(k)$ that satisfies the dynamical constraints of the internal model (1). Thus, *Stage 2* is enabled based on the parameters $\alpha(k)$ and $\tilde{x}(k)$.

Step 2: Convex Decomposition

The regulation problem around the virtual reference is addressed through an IBC applied to the error between the current state and the virtual state (4.16). The optimization performs the convex decomposition of the error between the re-scaled outer set $\alpha(k)\Omega^v$ and the re-scaled inner set $\alpha(k)\Omega^o$.

$$\begin{bmatrix} \varepsilon^v(k) \\ \varepsilon^o(k) \\ c(k) \end{bmatrix} = \underset{(\varepsilon^v(k),\varepsilon^o(k),c(k))}{\arg\min} \quad c(k)$$
$$\text{subject to} \quad \begin{aligned} &\varepsilon^v(k) \in \alpha(k)\Omega^v, \\ &\varepsilon^o(k) \in \alpha(k)\Omega^o, \\ &\varepsilon(k) = c(k)\varepsilon^v(k) + (1-c(k))\varepsilon^o(k), \\ &c(k) \in [0,1] \end{aligned} \tag{4.18}$$

The optimization (4.18) provides regulation errors in inner $\varepsilon^o(k) \in \alpha(k)\Omega^o$ and outer sets $\varepsilon^v(k) \in \alpha(k)\Omega^v$ and the convex factor $c(k)$ which defines which regulation action is preponderant.

Step 3: Convex Composition

The control action at step k is computed with the following formula:

$$u(k) = \tilde{u}(k) + \underbrace{c(k)v^v(\varepsilon^v(k)) + (1-c(k))v^o(\varepsilon^o(k))}_{v(k)} \tag{4.19}$$

where $v^v(k)$ and $v^o(k)$ are control actions that leaves $\alpha(k)\Omega^v$ and $\alpha(k)\Omega^o$ invariant (those actions exist due to the controlled invariance properties). Thus $u(k)$ is applied to the system (4.1) to compute $x(k+1)$. Then, the IBT will be implemented with the 3 stages procedure.

4.2.5 Properties

Preliminaries

First, let us recall a characterization of convex sets based on Minkowski sums that will be used further:

Lemma 4.2: Convex sets and Minkowski sum

A set C is convex if and only if :

$$\lambda C \oplus (1-\lambda)C = C, \forall \lambda \in [0,1] \tag{4.20}$$

Proof

$$\forall \lambda \in [0,1]\ \lambda C \oplus (1-\lambda)C = \{\lambda x + (1-\lambda)y \ \mid \ (x,y) \in C\} = C$$

□

Lemma 4.3: Convex polyhedron and Minkowski sum

Let C a convex polyhedron , $x \in \mathbb{R}^n$ and $\alpha \in [0,1]$.

$$\{x\} \oplus \alpha C \subset C \iff x \in (1-\alpha)C$$

Proof

$$\{x \in \mathbb{R}^n \mid \{x\} \oplus \alpha C \subset C\} = C \ominus \alpha C$$

and according to the previous characterization :

$$\begin{aligned} C \ominus \alpha C &= [(1-\alpha)C \oplus \alpha C] \ominus \alpha C \\ &= (1-\alpha)C \oplus \alpha C \ominus \alpha C \\ &= (1-\alpha)C \end{aligned}$$

Recursive feasibility

Once the methodology proposed for the Interpolation-based Tracking is clarified (in terms of the optimization to be solved in real-time and the construction of the set-parametrizations as well as the control action), we can concentrate on the analysis of the closed loop properties according to the classical desiderata for any recursive optimization-based strategy.

Lemma 4.4

For all $x_0 \in \Omega^v$ there exists $\tilde{x}_0 \in \Omega^v$ such that (4.17) is feasible.

Proof

We observe that there exist at least two feasible choices $\tilde{x}_0 = 0$ or $\tilde{x}_0 = x_0$. If $\tilde{x}_0 = 0$, then $\alpha = 1$ and any $\tilde{u}_0 \in \mathcal{U}$ is a feasible choice. If $\tilde{x}_0 = x_0$, then any $\alpha \in [0,1]$ and $\tilde{u}_0 = 0$ is a feasible choice.

This first result is particularly important as long as it offers a safe basic solution whenever the reference trajectory to be tracked is evolving dynamically. In other words, one can always choose the origin as a feasible virtual reference to be tracked.

Proposition 4.1

Given $x(k), \tilde{x}(k)$ and $x^{ref}(k)$, if (4.17) is feasible and

$$x(k+1) \in \{\tilde{x}(k+1|k)\} \oplus \alpha(k)\Omega^v, \tag{4.21}$$

then (4.17) is feasible for the pair of parameters $x(k+1|k), \tilde{x}(k+1|k)$ found as optimal solutions for (4.17) at step k and any $x^{ref}(k+1)$.

Proof

Assume (4.17) is feasible at step k and

$$x(k+1) \in \{\tilde{x}(k+1|k)\} \oplus \alpha(k)\Omega^v$$

By considering the feasible choice: $\tilde{x}(k+1|k+1) = \tilde{x}(k+1|k)$, the feasible choice $\alpha(k+1) = \alpha(k) \in [0,1]$ can be considered.

$$\{\tilde{x}(k+1|k)\} \oplus \alpha(k)\Omega^v \subset \Omega^v \Rightarrow \{\tilde{x}(k+1|k+1)\} \oplus \alpha(k+1)\Omega^v \subset \Omega^v$$

According to Lemma 4.3:

$$\{\tilde{x}(k+1|k+1)\} \oplus \alpha(k+1)\Omega^v \subset \Omega^v \Rightarrow \tilde{x}(k+1|k+1) \in (1-\alpha(k+1))\Omega^v$$

$\tilde{x}(k+1|k+1) \in (1-\alpha(k+1))\Omega^v$, so there exists $\tilde{u} \in (1-\alpha(k+1))\mathcal{U}$ such that:

$$\tilde{x}_{k+2|k+1} = A\tilde{x}(k+1|k+1) + B\tilde{u} \in (1-\alpha(k+1))\Omega^v$$

Let $\tilde{u}(k) \in (1-\alpha(k+1))\mathcal{U}$ such that

$$\tilde{x}_{k+2|k+1} = A\tilde{x}(k+1|k+1) + B\tilde{u} \in (1-\alpha(k+1))\Omega^v$$

According to Lemma 4.3:

$$\tilde{x}_{k+2|k+1} \in (1-\alpha(k+1))\Omega^v \Rightarrow \{\tilde{x}_{k+2|k+1}\} \oplus \alpha(k+1)\Omega^v \subset \Omega^v$$

□

With this result, a step forward is made toward a recursive construction and re-

utilisation of the previous feasible solution. What is particularly important here is that the feasibility of the optimization problem at time step $k+1$ is independent of the reference signal to be tracked. This last one is considered in the cost functions but is not linked to a constraint that can be rendered unfeasible.

Proposition 4.2

If (4.17) is feasible, then (4.18) is feasible for the respective solutions $\alpha(k)$ and $\tilde{u}(k)$.

Proof

$$\begin{aligned}\varepsilon(k+1) &= x(k+1) - \tilde{x}(k+1|k+1)\\ &= A(x(k+1) - \tilde{x}(k|k+1)) + B(u(k) - \tilde{u}(k))\\ &= A\varepsilon(k) + Bv(k)\\ &= A(c(k)\varepsilon^v(k) + (1-c(k))\varepsilon^o(k)) + B(c(k)v^v(k) + (1-c(k))v^o(k))\\ &= c(k)(A\varepsilon^v(k) + Bv^v(k)) + (1-c(k))(A\varepsilon^o(k) + Bv^o(k))\end{aligned}$$

$A\varepsilon^v(k) + Bv^v(k) \in \alpha(k)\Omega^v$ according to the invariance of $\alpha(k)\Omega^v$ at step k and $A\varepsilon^o(k) + Bv^o(k) \in \alpha(k)\Omega^o$. We choose:

$$\begin{aligned}c(k+1) &= c(k),\\ \varepsilon^v(k+1) &= A\varepsilon^v(k) + Bv^v(k),\\ \varepsilon^o(k+1) &= A\varepsilon^o(k) + Bv^o(k).\end{aligned}$$

□

Proposition 4.3

If (4.17) and (4.18) are feasible, then $u(k)$ computed in (4.19) satisfies $u(k) \in \mathcal{U}$ and the condition $x(k+1) \in \{\tilde{x}(k+1|k)\} \oplus \alpha(k)\Omega^v$ holds.

Proof

$u^v(k)(\varepsilon^v(k)) \in \alpha\mathcal{U}$ and $u^o(k)(\varepsilon^o(k)) \in \alpha\mathcal{U}$ according to Lemma 4.1. Then their convex combination : $c(k)u^v(k)(\varepsilon^v(k)) + (1-c(k))u^o(k)(\varepsilon^o(k)) \in \alpha\mathcal{U}$. Thus, $u(k)$ is a combination of two elements of $\alpha(k)\mathbb{U}$ and $(1-\alpha(k))\mathbb{U}$, so

$u(k) \in \mathcal{U}$.

$$\begin{aligned} x(k+1) - \tilde{x}(k+1) &= Ax(k) + Bu(k) - (A\tilde{x}(k) + B\tilde{u}(k)) \\ &= A(x(k) - \tilde{x}(k)) + \\ &\quad + B(\tilde{u}(k) + c(k)v^v(k) + (1 - c(k))v^o(k)) - B\tilde{u}(k) \\ &= A(x(k) - \tilde{x}(k)) + B(c(k)v^v(k) + (1 - c(k))v^o(k)) \end{aligned}$$

Due to constraints (4.17) : $(x(k) - \tilde{x}(k)) \in \alpha(k)\Omega^v$. Additionally the control action $c(k)v^v(k) + (1 - c(k))v^o(k)$ is constructed to leave the current error invariant in $\alpha(k)\Omega^v$. Consequently $(x(k+1) - \tilde{x}(k+1)) \in \alpha(k)\Omega^v$. □

These last two results allow to guarantee that the feasibility is not lost at the stage 2 and respectively 3 of the IBT procedure and enables the statement of the next results.

Proposition 4.4

The IBC for tracking procedure is recursively feasible.

Proof

Lemma 4.4 guarantees the initialization of optimization (4.17). Let (4.17) be feasible at step k. Then (4.18) is feasible at step k according to Proposition 4.2 and it implies that the condition: $x(k+1) \in \{\tilde{x}(k+1|k)\} \oplus \alpha(k)\Omega^v$ holds thanks to Proposition 4.3. Consequently, (4.17) is feasible at step $k+1$ thanks to Proposition 4.1. □

Practical Implementation and discussion

The optimization (4.17) is a QP problem if Ω^v and Ω^o are polyhedrons and the complexity of the optimization arguments is $(m+1)$ and (4.18) is a bilinear programming problem of complexity $(2m+1)$ that can be rewritten in terms of a LP problem following the transformation based on the change of variable $r^v(k) = c(k)\varepsilon^v(k)$ as showed in Section 4.1.4.

The choice of the initial virtual state $\tilde{x}_0$ is important for the behavior of the system, if $\tilde{x}_0 = 0$ then, the optimization problem (4.17) has more degrees of freedom to choose a higher scaling factor $\alpha(k)$ and consequently to make the regulation be preponderant at the beginning until the trajectory of the system (1) reaches the virtual reference where $\alpha(k)$ is close to 0. If $\tilde{x}_0 = x_0$ then the optimization (4.17) is always preponderant if there is no perturbation. It can

be noted that there is no weighting on the control action $\tilde{u}(k)$ as this would be redundant with the constraint.

Few remarks can be done with respect to qualities of the generic tracking method based on the IBT and the conservativeness of the proposed approach. First of all, the assumptions on convexity and most importantly on the one-step positive invariance of inner and outer sets seem difficult to drop without canceling the structural properties of the optimization problems involved in the IBT. While this is true for the convexity, it will be shown in the next chapter that the rigid one-step invariance can be relaxed opening the way to a wide range of alternative construction.

If we focus on the inner and outer sets that have been used in the constructions of this chapter, one can remark that their topology doesn't change and the optimization problems are handling only translations and scalings. The flexibility and the performances of the tracking policy can be enhanced if the sets are constructed at each step by optimizing their shape. Such constructions are computationally intensive although less conservative. They either involve set iterations or bring the optimization towards the nonlinear programming framework which deserve particular care from the global optimality and the computational load.

The constraints and the optimization complexity depend directly on the representation of the outer set. In the present work, the outer set is polyhedral, as linear constraints account to the number of half-spaces of the outer set. We stress the fact that for real-time systems, identifying the subset of active constraints can bring an important decrease of complexity for the optimization. If the reference signal is know in advance, such a pre-positioning of the constraints can be sought.

4.2.6 Optimizing the virtual feasible reference

In this subsection, we propose an extended formulation for the reference governor (4.17) by optimizing the virtual state $\tilde{x}$ at the same time with the control action $\tilde{u}$ and the scaling factor α. Consequently, the reference governor problem becomes:

$$\begin{aligned}
\underset{(\tilde{x(k)},\tilde{u}(k),\alpha(k))}{\text{minimize}} \quad & \|x^{ref}(k+1) - \tilde{x}(k+1|k)\|_Q^2 \\
\text{subject to} \quad & \tilde{x}(k+1|k) = A\tilde{x}(k|k) + B\tilde{u}(k), \\
& x(k) \in \{\tilde{x}(k|k)\} \oplus \alpha(k)\Omega^v, \\
& \{\tilde{x}(k|k)\} \oplus \alpha(k)\Omega^v \subset \Omega^v, \\
& \tilde{u}(k) \in (1-\alpha(k))\mathcal{U}, \\
& \{\tilde{x}(k+1|k)\} \oplus \alpha(k)\Omega^v \subset \Omega^v
\end{aligned} \tag{4.22}$$

Theorem 4.5

The tracking procedure based on optimization (4.22) is recursively feasible.

Proof

The proof is the same as the one provided for the previous procedure. □

By finding the virtual state as an optimization variable in the problem (4.17), the procedure resets $\tilde{x}(k)$ at each time step and thus the trajectory is less conservative and the controller more aggressive in the sense that arbitrary changes in the reference will be transmitted in the virtual reference.

Example: IBT

Consider the discrete-time linear system:

$$x(k+1) = \begin{bmatrix} 1 & 0.08 \\ 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} 0.0032 \\ 0.08 \end{bmatrix} u(k) \tag{4.23}$$

$$\text{subject to: } \begin{cases} -2.6 \leq x_1 \leq 2.6 \\ -3 \leq x_2 \leq 3 \\ -5 \leq u \leq 5 \end{cases}$$

We consider, as the inner set Ω^o, the maximal admissible positively invariant set with respect to the linear feedback law:

$$u(k) = -\begin{bmatrix} 16.6529 & 6.2331 \end{bmatrix} x(k) \; \forall k \in \mathbb{N} \tag{4.24}$$

We consider as the outer set a N-Step controlled invariant set reaching to Ω^o $C_N(\Omega^o)$ associated to a Vertex control law (4.8)
The ideal reference is a trajectory that leaves the state constraints set. The initial state of the virtual admissible reference is set to the origin. The initial state of the system is on the frontier of the controlled invariant $C_N(\Omega^o)$ at a vertex state with a zero initial speed. The IBT is implemented according to the procedure provided in the Section 4.2.3
Figures 4.9-4.10-4.11 present the simulations obtained for the numerical model. Figure 4.10 presents the controlled invariant $C_N(\Omega^o)$ and the relative positions of the reference signal, virtual reference and the state trajectory. Figure 4.11 details the time dependence for the IBT and the extended IBT strategies allowing to observe the controller first finds a scaling factor $\alpha_0 = 1$. This can be understood by the fact that the control action is equivalent to the one of a IBC regulation to the origin. Another natural conclusion is that higher the scaling factor, more conservative the reference governor through the virtual reference.
Whenever the regulation manages to lead the system close to the virtual reference, the scaling factor decreases and the reference governor generates the full control input and coincides with the system state. Reference governor control part is scaled by $(1-\alpha(k))\mathcal{U}$ and the IBC part is scaled by $\alpha(k)\mathcal{U}$.

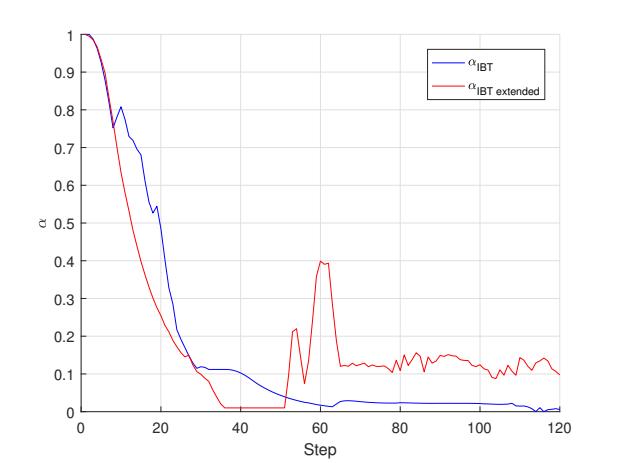

Figure 4.9: Scaling factor α for IBT (blue) and for extended IBT (red)

For the standard IBT, the virtual trajectory tends to move away from the frontier due to the conservativeness induced on one hand, by the re-scaling of the outer set instead of its reconstruction and, on the other hand by the dynamics imposed to the whole trajectory. The dynamics of the extended IBT solution is less constrained by the reset of the virtual state $\tilde{x}$ at each step. However, the structural properties of the scheme is exclusively based on convex (LP/QP) optimization and thus the computational performances prove to be very attractive.

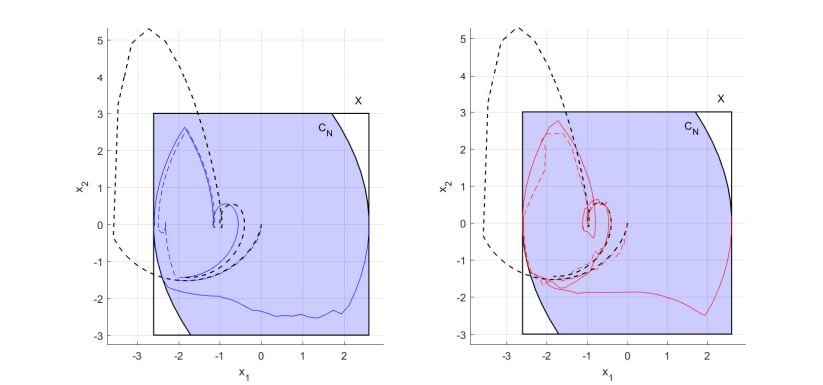

Figure 4.10: Left : Trajectories for the IBT of the real reference x^{ref} (black, dashed),the virtual reference $\tilde{x}$ (blue, dashed) and the system x (blue), Right : Trajectories for the extended IBT of the real reference x^{ref} (black, dashed),the virtual reference $\tilde{x}$ (red, dashed) and the system x (red)

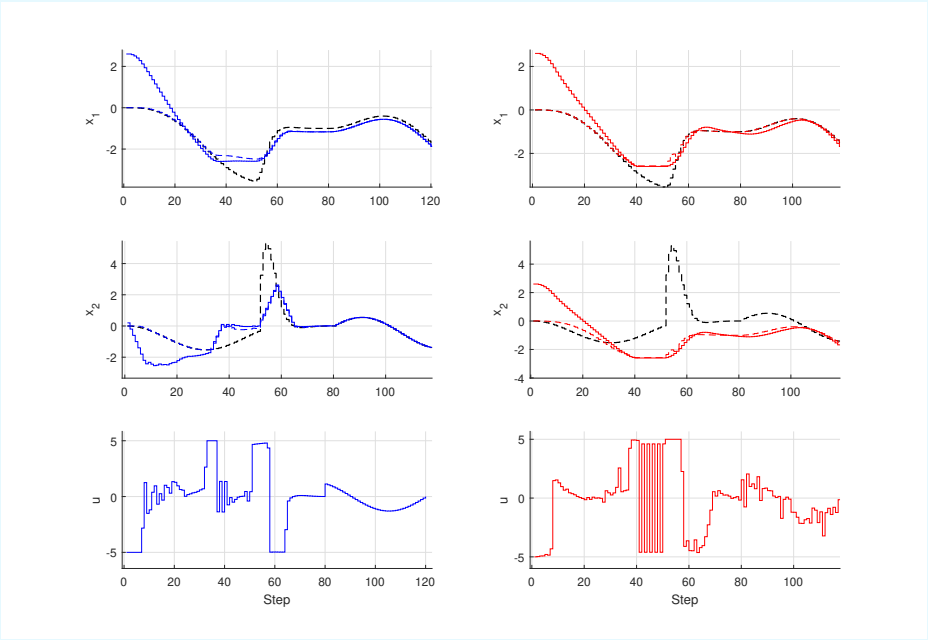

Figure 4.11: Left column : state and input trajectories for IBT, Right column: state and input trajectories for extended IBT

4.3 Interpolation Based Tracking for MCA

In this section we get back to the core application of the thesis and describe the IBT strategy in this framework. The particularity of this application would be the cost function which includes the washout component and the fact that the future trajectory to be tracked is not necessarily known in advance as it is the case for tracking MPC strategy. This last feature has advantages and disadvantages. On the positive side, we obtain a MCA control algorithm which is reactive with respect to the short term accelerations to be tracked thus avoiding delays and consequently the motion sickness. On the disadvantage one can count an activation of the position constraints at a relative late stage, which is inherent to the lack of anticipation, which is proper to MPC.

The enhancement of MCAs in order to improve the performance of acceleration rendering during the trajectory tracking in optimization-based MCA can benefit from the interpolation-based tracking strategies. Classical optimization-based algorithms generate delays due to computational complexity, particularly when an anticipation is hard to provide for high frequencies reference signals. The IBT handles low complexity optimization problems by guaranteeing recursive feasibil-

ity and stability.

This IBT technique needs less parameters to tune and a significative part of operations are done offline such as construction of the maximal controllable set which is essential ingredient to guarantee recursive feasibility of optimization problems and to exploit additional degrees of freedom as possible. The main conceptual step forward is that MCA handles real-time information during the simulation to perform the acceleration tracking by avoiding constraints manipulations on a long prediction horizon.

In this section we only consider the rail dynamics and we recall its model (See Section 2.2.2) below:

$$x(k+1) = \begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} \frac{1}{2}T_s^2 \\ T_s \end{bmatrix} u(k) \tag{4.25}$$

where T_s denotes the sampling time. At each sampling time k, $x(k) = \begin{bmatrix} p(k) & v(k) \end{bmatrix}^\intercal$ denotes the vector of position $p(k)$ and speed $v(k)$ while $u(k) = a(k)$ denotes the acceleration of the system according to a direction. Consider an acceleration reference u_{ref}, then a classical formulation for model predictive controller for MCA with respect to model (4.25) would be:

$$\begin{aligned} \underset{\substack{u(k),\dots,\\ u(k+N_h-1)}}{\text{minimize}} \quad & \sum_{k=0}^{N_h} \|u_{ref}(k) - u(k)\|_R^2 + \|x(k)\|_{Q_x}^2 + \|x(N_h)\|_P^2 \\ \text{subject to} \quad & x(k) \in \mathcal{X},\ \forall k \in \{1,\dots,N_h\}, \\ & u(k) \in \mathcal{U},\ \forall k \in \{0,\dots,N_h-1\}, \\ & x(N_h) \in \mathcal{X}_f \end{aligned} \tag{4.26}$$

However, in the driving simulation the reference acceleration profile can be a priori unknown and the performance of the MPC controller increases with the prediction horizon N_h, consequently the controller is not practically adapted to small prediction horizons. Moreover the trade-off between prediction length and complexity of the optimization problems implies performing MPC controllers have to handle many constraints and thus impact the optimization solving routine.

Example: IBT-MCA

In this section, we consider the lateral acceleration rendering during a slalom phase with respect to two MCAs:

1. MPC-MCA (4.26) with a prediction horizon of 3.2 seconds.
2. IBT-MCA with a convex polyhedral outer set Ω^v built as a N-step controllable set reaching $\mathcal{X}_f$.

Figure 4.12 (resp Figure 4.13) depicts the state space trajectory of the system with respect to the IBT-MCA (resp MPC-MCA), The largest controlled invariant set C_N is represented in blue and the outer set Ω^v in red. The better exploitation of the state space of the IBT-MCA can be noticed

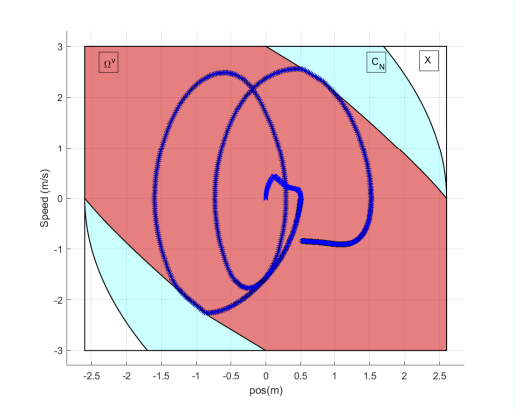

Figure 4.12: State-space trajectory for IBT

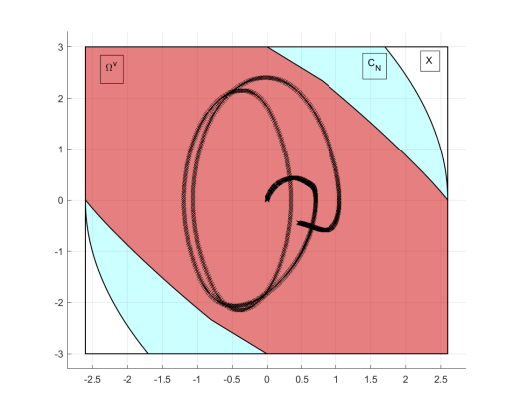

Figure 4.13: State-space trajectory for MPC

The comparison of acceleration renderings is depicted on Figure 4.14, during the slalom. The acceleration rendering of IBT appears to reproduce better the shape of the reference particularly during the period of variation change (from 4.5 to 5s).

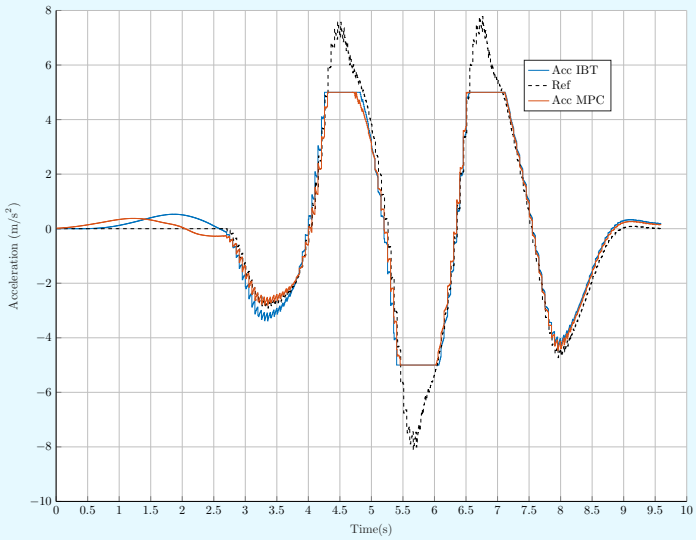

Figure 4.14: Comparison of acceleration rendering for IBT and MPC

Chapter 5

p-Invariance

Contents

In the field of constrained control, the Dual Mode paradigm allowed the emergence of efficient control techniques based on finite-dimensional real-time optimization for constrained dynamical systems. The Dual Mode paradigm consists in making the states of a system evolve along a trajectory within a feasible or controllable set until they reach a positively invariant terminal set associated to a stabilizing linear local control law [Mayne et al., 2000].

Thus, Model Predictive Control (MPC) applies this scheme on a receding horizon in order to guarantee the recursive feasibility of the closed loop. It should be noted that this paradigm is also implicitly embedded in alternative techniques such as Interpolation-Based Control [Nguyen et al., 2011a, Nguyen et al., 2013, Nguyen, 2014] which do not need a prediction horizon and has comparable performance as MPC with less computational load. Indeed, one of the interpolation terms is a "local" control invariant set which practically plays the role of a terminal set in dual-mode MPC.

Many works give methods to build positively invariant sets for discrete time linear systems [Gutman and Cwikel, 1986, Gilbert et al., 1995], their characterization with Farkas lemma [Bitsoris, 1988, Dorea and Hennet, 1999], extension to uncertain systems [Blanchini and Miani, 2008] and this field is actually in development nowadays for nonlinear systems.

However, those sets on one hand may be difficult to compute and on the other hand can be difficult to use in optimization problems due to their complexity. More complex the positively invariant set is, the more constraints optimization solver has to handle. Finally lesser volume of the positively invariant set within the feasible set tends to make the controlled dynamics more conservative in terms of domain of attraction for the closed loop.

A solution to those drawbacks was to consider simpler sets with a priori given complexity (or polyhedral approximation of positively invariant set) [Athanasopoulos et al., 2014, Hovd et al., 2014, Scibilia et al., 2011],as terminal sets. Another approach relaxed the positive invariance properties and thus may allow a trajectory to leave for some instants the terminal set before returning in it which define one of the first notions of periodic invariance [Lee, 2004, Lee and Kouvaritakis, 2006]. The stability and dual-mode principle can still be used based on such constructions as long as the terminal set is in the interior of a controlled invariant set.

This chapter deals with p-invariant sets and aims to introduce new concepts, to formally describe their properties and point to the constructive procedures. Their potential of computation load alleviation for optimization problem solving is one of the motivation for the analysis of this concept.

Moreover, we consider two notions of p-invariance,

- weak invariance that offers new perspectives of validation of static constraints
- a strong invariance which is more restrictive but easier to use for control design.

In order to introduce those properties of p-invariance we propose firstly adress the weak satisfaction of constraints that is the case where a trajectory is allowed to violate a constraint during a finite time sequence. This weak satisfaction of

a constraint by a trajectory or a tube of trajectories leads to the notion of weak p-invariance for a set. Once such a property is established, it guarantees that every trajectories initiated in such a set returns in it in a finite time window.

A strong version of constraint satisfaction and invariance can be derived from the weak ones. Thus, a trajectory or a tube of trajectories starting in the considered set has to return in it before a finite time.

In a first part we will consider the notions of p-invariance for autonomous systems by introducing tools for analysis of such configurations. Then we will develop to what extent those concepts can be used to controllable systems and underlined properties and applications.

5.1 p-invariance for autonomous systems

Let us focus on autonomous discrete-time systems whose model is recalled below:

$$x(k+1) = f(x(k)) \tag{5.1}$$

5.1.1 p-satisfaction of constraints

For a continuous convex function $h : \mathbb{R}^n \to \mathbb{R}$, we define the sublevel set of h as follows [1]:

$$\mathcal{L}(h) = \{x \in \mathbb{R}^n \mid h(x) \leq 0\} \tag{5.2}$$

The main goal is to study the set-membership for a trajectory of (5.1) with respect to the sublevel set. In this purpose we define the *Validation Index Set*.

Definition 5.1: Validation Index Set

A *Validation Index Set* (VIS) $\mathcal{V}(x_0, f, h) \subset \mathbb{N}$ is an ordered collection of indices $i \in \mathbb{N}$ whose states $x(i)$ belong to the sublevel set $\mathcal{L}(h)$.

[1] In this work we will refer indifferently to satisfaction of the constraint $h(x) \leq 0$ and the inclusion $x \in \mathcal{L}(h)$ through *the satisfaction of* $\mathcal{L}(h)$

Definition 5.2: Maximal Validation Index Set

The *maximal Validation Index Set* (mVIS) is the maximal ordered collection of indices VIS (w.r.t set inclusion). It can be formally described as:

$$\mathcal{V}^m(x_0, f, h) = \left\{ t \in \mathbb{N} \mid x(t) = f^t(x_0) \in \mathcal{L}(h) \right\} \tag{5.3}$$

where f^t is the power t^{th} power of f *i.e* $f^t(x) = \underbrace{f(f(f(...)))}_{t \text{ times}}(x)$.

where we recall that $f(.)$ is single value

Assume the autonomous dynamic system (5.1), a constraint modeled by a sublevel set $\mathcal{L}(h)$ (5.2), and let us focus on the case for which a trajectory violates the constraint for a finite time interval before satisfying it again.

Definition 5.3: Weak p-satisfaction of constraint

Given finite $p \in \mathbb{N}^*$, the trajectory of the system (5.1) initiated in $x_0 \in \mathbb{R}^n$ is weakly p-satisfying the constraint $\mathcal{L}(h)$ if it exists a function $r : \mathbb{N} \to \mathbb{N}_{[1,p]}$ such that $x(k + r(k)) \in \mathcal{L}(h)$ for any $k \in \mathbb{N}$.

In other words, a trajectory weakly p-satisfies a constraint if we can assure it reaches $\mathcal{L}(h)$ at least once during any time interval of length p. This notion is illustrated on Fig. 5.1 for a generic trajectory.

We provide a necessary and sufficient condition of weakly p-satisfaction of a trajectory with respect to a constraint.

Theorem 5.1

Given a finite $p \in \mathbb{N}^*$, the trajectory of the system (5.1) initiated in x_0 weakly p-satisfies the constraint $\mathcal{L}(h)$ if and only if :

- the *mVIS* $\mathcal{V}^m(x_0, f, h)$ is unbounded
- The difference between two consecutive elements is bounded and:

$$\begin{aligned} \sigma = \max_{j \in \mathbb{N}} \quad & t_{j+1} - t_j \\ \text{s.t.} \quad & (t_j, t_{j+1}) \in \mathcal{V}^m(x_0, f, h) \end{aligned} \tag{5.4}$$

- $\sigma \leq p$

Proof

If the trajectory weakly p-satisfies $\mathcal{L}(h)$ then there exists a function $r : \mathbb{N} \to \mathbb{N}_{[1,p]}$ such that

$$x(k + r(k)) \in \mathcal{L}(h) \; \forall k \in \mathbb{N}$$

and the *mVIS* verifies:

$$\begin{array}{l} \mathcal{S} = \{r(0), r(0) + r(r(0)), \\ r(0) + r^2(0) + r(r(0) + r^2(0)), \dots \} \subset \mathcal{V}^m(x_0, f, h) \end{array}$$

$\mathcal{S}$ being an nonempty and unbounded subset of $\mathcal{V}^m(x_0, f, h)$, the optimization problem is feasible (any element $t_j \in \mathcal{V}^m(x_0, f, h)$ has a successor $t_{j+1} \in \mathcal{V}^m(x_0, f, h)$ in this ordered set).
Consider any pair of successive points $(t_j, t_{j+1}) \in \mathcal{V}^m(x_0, f, h) \times \mathcal{V}^m(x_0, f, h)$ with $t_j \geq r(0)$ it exists a pair of successive points $(s_i, s_{i+1}) \in \mathcal{S} \times \mathcal{S}$ such that $s_i \leq t_j \leq t_{j+1} \leq s_{i+1}$.

$$\begin{array}{rcl} t_{j+1} - t_j & \leq & s_{i+1} - s_i \\ & \leq & r(s_i) \\ & \leq & p \end{array}$$

For any pair of successive points $(t_j, t_{j+1}) \in \mathcal{V}^m(x_0, f, h) \times \mathcal{V}^m(x_0, f, h)$ with $t_j < r(0)$ we have $t_{j+1} \leq r(0)$ because $\mathcal{V}^m(x_0, f, h)$ is the *maximal* VIS. It follows that:

$$t_{j+1} - t_j \;\; \leq \;\; r(0) - t_j \leq r(0) \leq p$$

For the sufficiency, one can note that $\mathcal{V}^m(x_0, f, h) = (t_i)_{i \in \mathbb{N}}$ being (ordered) unbounded set then one can define the function :

$$\begin{array}{rcl} r : \mathbb{N} & \longrightarrow & \mathbb{N}_{[1,p]} \\ r(k) & = & \left\{ \begin{array}{ll} t_0 - k & \text{if } k < t_0 \\ t_{j+1} - k & \text{if } t_j \leq k < t_{j+1} \end{array} \right. \end{array}$$

thus $k + r(k) \in \mathcal{V}^m(x_0, f, h)$ and the trajectory $(x(k))_{k \in \mathbb{N}}$ is weakly p-satisfying the constraint according to Def. 5.3. □

In other words, a trajectory weakly p-satisfies a constraints if and only if two successive elements of the *mVIS* are separated by a distance bounded by p.

Notions of *VIS* and weak p-satisfaction can be extended to configuration with vector constraint $h^m : \mathbb{R}^n \to \mathbb{R}^m$, where $h^m = [h_1^m, \dots, h_m^m]^\intercal$

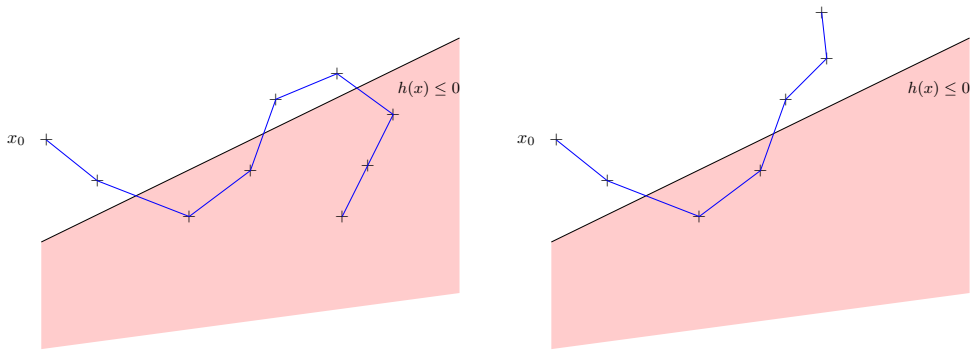

Figure 5.1: Left: the trajectory satisfies the constraint $h(x) \leq 0$ with an index 3. Right: No p-satisfaction of constraint

Definition 5.4: Weak vector-constraint satisfaction

A trajectory of the system (5.1) initiated at $x_0 \in \mathbb{R}^n$ weakly p-satisfies the vector constraint $\mathcal{L}(h^m)$ if it exists a function $r : \mathbb{N} \to \mathbb{N}^m_{[1,p]}$ such that

$$x(k + r_i(k)) \in \mathcal{L}(h^m_i), \forall i \in \{1, \ldots, m\}$$

for any $k \in \mathbb{N}$ where $r_i(.)$ is the i-th component of r.

Theorem 5.2

The trajectory of the system (A.25) weakly satisfies the $m-$dimensional vector-constraint $\mathcal{L}(h^m)$ if and only if

- All the *VIS* $\mathcal{V}(x_0, f, h^m_i)$ with $i \in \mathbb{N}_{[1,m]}$ are unbounded;
- The following optimization problem has a bounded solution:

$$\begin{array}{ll} \max\limits_{i \in \mathbb{N}_{[1,m]}} & \max\limits_{j \in \mathbb{N}} \{t_{j+1} - t_j\} \\ \text{s.t.} & t_j \in \mathcal{V}^m(x_0, f, h^m_i) \end{array} \tag{5.5}$$

Proof

According to Theorem 5.1 applied here for each $i \in \mathbb{N}_{[1,m]}$, the constraint $\mathcal{L}(h^m_i)$ is weakly satisfied by the trajectory if and only if the optimization

problem (5.4) has bounded solution.
As a consequence, the optimization (5.5) is feasible and has a bounded solution given by the most conservative index of weak satisfaction. □

Remark. *The weak satisfaction of the multiple constraints doesn't guarantee the simultaneous satisfaction of constraints on a finite time interval. The Fig. 5.2 illustrates this feature using a dynamics provided in [La Salle, 1976]:*

$$x(k+1) = \begin{bmatrix} \dfrac{ax_2(k)}{1+x_1^2(k)} \\ \dfrac{bx_1(k)}{1+x_2^2(k)} \end{bmatrix} = f(x(k)) \tag{5.6}$$

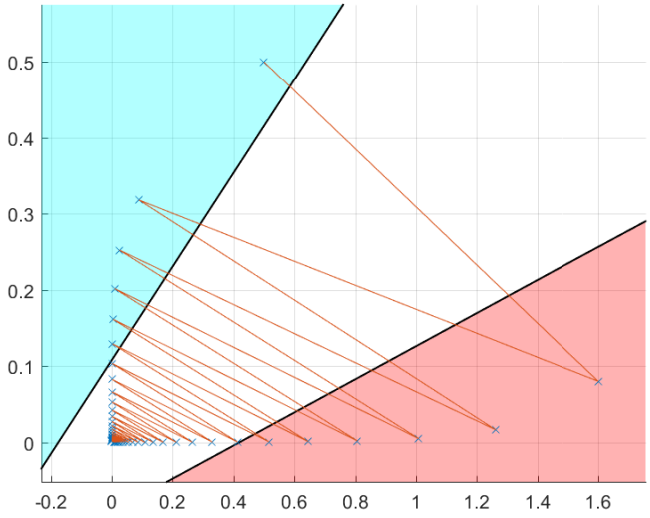

Figure 5.2: Non-simultaneous satisfaction of constraints together with the validation of a weak satisfaction of constraints with a index 2.

The same notions can be also extended to tube of trajectories.

Definition 5.5: Validation Index Set for tube of trajectories

The *mVIS* defined in Definition 5.1 with respect to a scalar constraint $\mathcal{L}(h)$ and the system (5.1) can be extended for tube of trajectories initiated in $X \subset \mathbb{R}^n$ with the following formulation :

$$\mathcal{V}^m(X, f, h) = \{t \in \mathbb{N} \mid f^t(X) \subset \mathcal{L}(h)\}. \tag{5.7}$$

In other words, for a given subset X of $\mathbb{R}^n$, the mVIS $\mathcal{V}^m(X, f, h)$ is an ordered subset of $\mathbb{N}$ which collects the indices of successive images of initial conditions in X via $f(.)$ such that the trajectories are simultaneously included in the sublevel set $\mathcal{L}^h$.

Proposition 5.1

The mVIS for tubes of trajectories is the intersection of mVIS for each individual trajectories within the tube:

$$\mathcal{V}(X, f, h) = \bigcap_{x \in X} \mathcal{V}(x, f, h) \tag{5.8}$$

Proof

$$\begin{aligned} V(X, f, h) &= \{t \in \mathbb{N} \mid f^t(X) \subset \mathcal{L}(h)\} \\ &= \{t \in \mathbb{N} \mid \forall x, \in X f^t(x) \in \mathcal{L}(h)\} \\ &= \bigcap_{x \in X} \{t \in \mathbb{N} \mid f^t(x) \in \mathcal{L}(h)\} \\ &= \bigcap_{x \in X} \mathcal{V}(x, f, h) \end{aligned}$$

□

Definition 5.6: Weak p-satisfaction of constraint for tubes

Given $p \in \mathbb{N}^*$, the trajectories of the system (5.1) initiated in $X \subset \mathbb{R}^n$ weakly p-satisfy the vector constraint $\mathcal{L}(h^m)$ if it exists a function $r : \mathbb{N} \to \mathbb{N}^m_{[1,p]}$ such that

$$x(k + r_i(k)) \in \mathcal{L}(h^m_i), \forall i \in \{1, \ldots, m\}$$

for all $x_0 \in X$ and for any $k \in \mathbb{N}$ where $r_i(.)$ is the i-th component of r.

Theorem 5.3

The tube of trajectories of the system (5.1) initiated in $X \subset \mathbb{R}^n$ weakly p-satisfies the $m-$dimensional vector-constraint $\mathcal{L}(h^m)$ if and only if

- All the m*VIS* $\mathcal{V}(X, f, h_i^m)$ with $i \in \mathbb{N}_{[1,m]}$ are unbounded;
- The following optimization problem has a bounded solution:

$$\begin{aligned} \max_{i \in \mathbb{N}_{[1,m]}} \quad & \max_{j \in \mathbb{N}} \{t_{j+1} - t_j\} \\ \text{s.t.} \quad & (t_j, t_{j+1}) \in \mathcal{V}^m(X, f, h_i^m) \end{aligned} \tag{5.9}$$

Proof

Theorem 5.3 is verified if and only if every trajectory initialised in X weakly $p-$satisfy the vector-constraint $\mathcal{L}(h^m)$ and consequently if and only if every trajectory verifies Theorem 5.2.

□

Up to this point, the set of initial conditions $X \subset \mathbb{R}^n$ was considered to be independent of the constraints whose satisfaction is under study i.e. $h^m(x) \leq 0$. In the case the set is defined as $X = \mathcal{L}(h^m)$ one can talk about the self p-satisfaction of constraints.

5.1.2 Weak p-invariance

In the previous section we defined notions of p-satisfaction of constraints for trajectories and tubes of trajectories which give rise to the definition of p-invariance properties for a set with respect to a dynamical system..

A set can be defined as an intersection of constraints and the weak p-satisfaction of each such constraint can be analysed in the previous framework. If this analysis is carried out with respect to initial states in the set itself we move from the satisfaction of constraints by a tube of trajectories towards the weak notion of p-invariance of a set.

First the notions are presented in a general nonlinear framework and subsequently, the properties are shown to lead to constructive indices in the LTI case considering polyhedral sets.

We define the weak p-invariance with the same tools than previous notion of constraints satisfaction:

Definition 5.7: Weak invariance

Let $p \in \mathbb{N}$. The set $\Omega \subset \mathbb{R}^n$ is weakly p-invariant with respect to the system (5.1) if for any $x_0 \in \Omega$, it exists a function $r : \mathbb{N} \to \mathbb{N}_{[1,p]}$ such that $x(k+r(k)) \in \Omega$ for any $k \in \mathbb{N}$.

Proposition 5.2

A set $\Omega \subset \mathbb{R}^n$ is weakly p-invariant with respect to the system (5.1) if and only if the constraint $h(x) \leq 0$ defined by the function:

$$\begin{array}{rcl} h_\Omega : \mathbb{R}^n & \longrightarrow & \mathbb{R} \\ h_\Omega(x) & = & \begin{cases} 1 & \text{if } x \notin \Omega \\ -1 & \text{if } x \in \Omega \end{cases} \end{array} \tag{5.10}$$

is weakly p-satisfied by the tube of trajectories initiated in Ω.

Theorem 5.4

A set $\Omega \subset \mathbb{R}^n$ containing the origin in its interior is weakly p-invariant with respect to the system (5.1) if and only if

- $\mathcal{V}(x_0, f, h_\Omega)$ is unbounded for all $x_0 \in \Omega$
- The optimization problem:

$$\begin{array}{rll} \sigma = \max\limits_{x_0 \in \Omega} & \max\limits_{j \in \mathbb{N}} \{t_{j+1} - t_j\} \\ \text{s.t.} & (t_j, t_{j+1}) \in \mathcal{V}^m(x_0, f, h_\Omega) \end{array} \tag{5.11}$$

  has a bounded solution.
- $\sigma \leq p$

Proof

If the set Ω is weakly p-invariant, the unboundedness of $\mathcal{V}^m(x_0, f, h)$ is verified for every $x_0 \in \Omega$ following the same arguments in Theorem 5.1. Consequently, the optimization problem is feasible and its solution is bounded by p thanks to the weak p-invariance of Ω. Conversely, the satisfaction of the first

conditions proves the feasibility of the optimization problem. By choosing p the optimal solution, the conditions of weak invariance are fulfilled. □

The following two corollaries are direct applications of the definition and the theorem above.

Corollary 5.1

If a set $\Omega \subset \mathbb{R}^n$ is weakly p-invariant with respect to the system (5.1) then it will be weakly p-invariant for all $p > p_0$.

Corollary 5.2

The weakly 1-invariance is equivalent to the classical notion of positive invariance (Definition 2.3).

The set Ω was used until now using the associated function h_Ω in (5.2) but its practical description is often done based on joint satisfaction of set of constraints:

$$\Omega = \bigcap_{i=1}^{N} \mathcal{L}(h_i) \tag{5.12}$$

The following proposition establishes a link between these two notions.

Proposition 5.3

If the set Ω defined as an intersection (5.12) is weakly p-invariant, then the tube of trajectories initiated in this set weakly p-satisfies the respective constraints $\mathcal{L}(h_i)$ $\forall i$.

The proof is a direct consequence of the fact that $h_i(x) \leq 0$ for all $x \in \Omega$ and thus for all x satisfying $h_\Omega(x) \leq 0$. What is important is to note that the converse is not true as exemplified graphically by the Figure 5.3.

Corollary 5.3

Let the set Ω defined as a finite intersection (5.12). If it exists $p \in \mathbb{N}$ such that Ω is weakly p-invariant, then the constraints $\mathcal{L}(h_i)$ are weakly p_i-satisfied

with

$$p \geq \max_{i=1\dots m} p_i \tag{5.13}$$

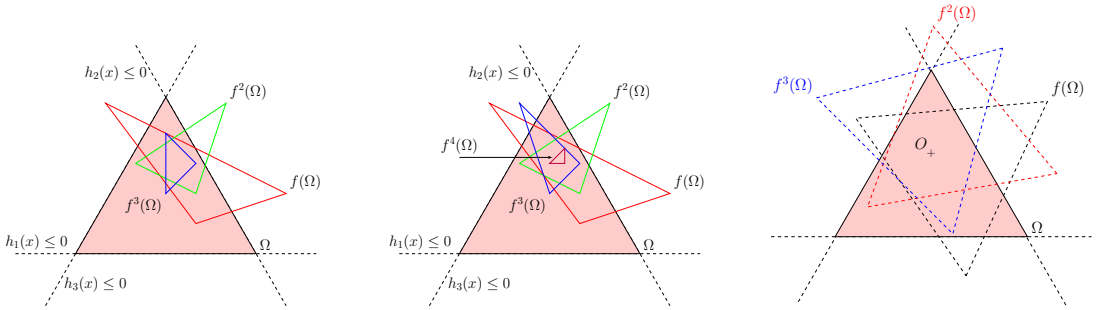

Figure 5.3: Left: $p = \max_i p_i$. Middle: $p > \max_i p_i$. Right: no weak p-invariance while every constraints are weakly p-satisfied

Theorem 5.5

Ω is weakly p-invariant w.r.t. (5.1) if, and only if:

$$\Omega \subset \bigcup_{i=1}^{p} f^{-i}(\Omega). \tag{5.14}$$

Proof

(IF) Condition (5.14) implies that, $\forall k \in \mathbb{N}$, if $x(k) \in \Omega \subset \bigcup_{i=1}^{p} f^{-i}(\Omega)$, then, there exists $1 \leq r(k) \leq p$ such that $x(k + r(k)) \in \Omega$. Since it is also true for $k = 0$, then, for any $x_0 = x(0) \in \Omega$, there exists $r(k)$ defined as $r : \mathbb{N} \to \mathbb{N}_{[1,p]}$ such that $x(k + r(k), x_0) \in \Omega$, which proves the sufficiency.
(ONLY IF): By contradiction, consider $x(0) = x_0 \in \Omega$, but $x_0 \notin \bigcup_{i=1}^{p} f^{-i}(\Omega)$. Then, $x(i) \notin \Omega \; \forall i = 1, \cdots, p$, implying that Ω does not satisfy the conditions stated in the definition of weak p-invariance. □

Weak invariance of polyhedral set with respect to LTI dynamics

As shown in the case of weak constraint satisfaction, whenever the dynamical system and the set of constraints has additional structural properties, the computation of the indices of weak invariance can be enhanced. The ultimate objective

of the present section is to present a commonly encountered case for which the computation is finitely determined.

Next we will detail the case of linear time-invariant systems:

$$x(k+1) = f_A(x(k)) = Ax(k) \tag{5.15}$$

with a Schur matrix A.

Theorem 5.6

Consider a compact set $\Omega \subset \mathbb{R}^n$ containing the origin in its interior. The minimal index p such that Ω is weakly $p-$invariant with respect to the asymptotically stable system (5.15) is finitely determined.

Proof

Given the asymptotic stability of the origin and the fact this is an interior point of Ω, there exist a finite time instant $\bar{p}$ such that

$$\mathbb{N}_{[\bar{p},\infty)} \subset \mathcal{V}^m(x_0, f, h_\Omega)$$

for all $x_0 \in \mathcal{V}(\Omega)$. Consequently the optimization (5.11) is finitely determined. □

The next results shows that homogeneity can be exploited in the verification of the weak invariance index.

Lemma 5.1

Consider a compact polyhedral set $\Omega \subset \mathbb{R}^n$ containing the origin in its interior. Let p be the minimal index such that Ω is weakly $p-$invariant with respect to the asymptotically stable system (5.15). Then p is the solution of the optimization problem:

$$\begin{aligned} p = \max_{x_0 \in \bar{\Omega}} \quad & \max_{j \in \mathbb{N}} \{t_{j+1} - t_j\} \\ \text{s.t.} \quad & (t_j, t_{j+1}) \in \mathcal{V}^m(x_0, f_A, h_\Omega) \end{aligned} \tag{5.16}$$

where $\bar{\Omega}$ is the boundary of Ω.

Proof

The Lemma claims that the index of weak invariance can be computed by optimizing over the constraint satisfaction indices of the trajectories initiated on $\bar{\Omega}$. By contradiction, suppose that the maximum index corresponds exclusively to an interior point $\tilde{x}$ such that $\tilde{x} \in \Omega$ and $f_A^p(\tilde{x}) \in \Omega$ but $f_A^i(\tilde{x}) \notin \Omega$ for all $i \in \mathbb{N}_{[1,p-1]}$.
Note however that there exist $\gamma > 1$ such that $\gamma\tilde{x} \in \bar{\Omega}$. By linearity of the dynamics and convexity of the set Ω it follows $\gamma f_A^i(\tilde{x}) \notin \Omega$ for all $i \in \mathbb{N}_{[1,p-1]}$. The constraint satisfaction index for the set Ω being p leads to

$$\gamma f_A^p(\tilde{x}) = f_A^p(\gamma\tilde{x}) \in \Omega$$

which shows that the optimum value of the optimization is obtained by the point $\gamma\tilde{x} \in \bar{\Omega}$ which leads to a contradiction. □

Despite the lack of relationship between the index of p-satisfaction of constraints for the vertices and the weak p-invariance index, the Theorem 5.5 and 5.6 can be exploited in order to obtain a finite index based on an explicit condition.

Corollary 5.4

A set Ω is weakly p-invariant w.r.t. (5.15) if and only if:

$$\Omega \subset \bigcup_{i=1}^{p} A^{-i}\Omega. \tag{5.17}$$

Moreover, if Ω is compact, then p is finite.

The condition (5.17) becomes particularly interesting if the set Ω is convex as long as the pre-image preserves this property and can lead to computationally friendly weak p-invariance test as for example: $\Omega = \bigcup_{i=1}^{p}(A^{-i}\Omega \cap \Omega)$ for which algorithmic procedures have been investigated in [Baotić, 2009].

5.1.3 Strong p-invariance

The weak constraint satisfaction and weak invariance offer two different perspectives on the validation of static constraints along the trajectories of a dynamical system. The present section aims to link the two notions through a *strong* version which imposes additional restrictions on the allowed interval between violation of constraints.

Definition 5.8: Strong vector-constraint p-satisfaction

The tube of trajectories of the system (5.1) initialized in $X \subset \mathbb{R}^n$ is strongly p-satisfying the vector constraints $\mathcal{L}(h)$ with $h : \mathbb{R}^n \to \mathbb{R}^m$ if $x(k+p) \in \mathcal{L}(h_i)$ $\forall i \in \mathbb{N}_{[1,m]}$, for any $x_0 \in X$ and for any $k \in \mathbb{N}$ such that $x(k) \in X$ and $x(k+1) \notin X$.

Theorem 5.7

The $m-$dimensional vector-constraint $h(x) \leq 0$ is strongly $p-$satisfied by the trajectory of the system (5.1) with initial condition $x_0 \in X \subset \mathbb{R}^n$ if and only if

- The sets $\mathcal{V}(x_0, h_i)$ with $i \in \mathbb{N}_{[1,m]}$ are unbounded;
- For each $i \in \mathbb{N}_{[1,m]}$ and for any successive elements $t_j \in \mathcal{V}(x_0, h_i)$ we have either $t_j + 1 \in \mathcal{V}(x_0, h_i)$ or $t_j + p \in \mathcal{V}(x_0, h_i)$.

Before establishing further results, we can notice that strong constrained satisfaction imposes stronger limitations on the set of validation indices and thus is expected to lead to larger values of p with respect to the weak counterpart. The next result stresses that strong notion covers the weak counterpart.

Corollary 5.5

If the $m-$dimensional vector-constraint $h(x) \leq 0$ is strongly $p-$satisfied by the trajectory of the system (5.1) with initial condition $x_0 \in X \subset \mathbb{R}^n$, then the same vector constraints are weakly $p-$satisfied.

Proof

By choosing $r = p$. □

Definition 5.9: Strong invariance

Let $p \in \mathbb{N}$. The set $\Omega \subset \mathbb{R}^n$ is strongly p-invariant with respect to the system (5.1) if $x(k+p) \in \Omega$ for each $x_0 \in \Omega$, and for all $k \in \mathbb{N}$ such that $x(k) \in \Omega$ and $x(k+1) \notin \Omega$.

This definition implies that a strong p-invariant set is also strong invariant for any multiples of p as index.

Corollary 5.6

If Ω is a strong p-invariant set, then it is also a strongly kp-invariant set for all $k \in \mathbb{N}^*$.

Theorem 5.8

A set $\Omega \subset \mathbb{R}^n$ containing the origin in its interior is strongly p-invariant with respect to the system (5.1) if and only if

$$\Omega \subset f^{-p}(\Omega) \cup f^{-1}(\Omega) \tag{5.18}$$

or equivalently

$$f(\Omega) \setminus \Omega \subset f^{-p+1}\Omega \tag{5.19}$$

Proof

The relationship (5.18) translates the p-invariance property in the set theoretic framework. For any initial condition $x \in \Omega$ two possibilities appear:

1. $f(x) \in \Omega$ then the strong p-invariance conditions are fulfilled.
2. $f(x) \notin \Omega$ then the trajectory has to reach Ω in $p-1$ steps.

Either $f(x) \in \Omega$ or alternatively $f(x) \notin \Omega$. For the first case, the strong p-invariance conditions are satisfied while in the second case, the trajectory needs to reach Ω in $p-1$ steps. Equivalently $f^{p-1}(f(\Omega) \setminus \Omega) \subset \Omega$. In the set theoretic framework, it becomes:

$$f(\Omega) \setminus \Omega \subset f^{-p+1}(\Omega) \tag{5.20}$$

□

Remark. *The strong p-invariance is close to the notion of (k, λ)-invariance defined in [Athanasopoulos et al., 2013] and [Lazar et al., 2013] in a stability analysis purpose.*

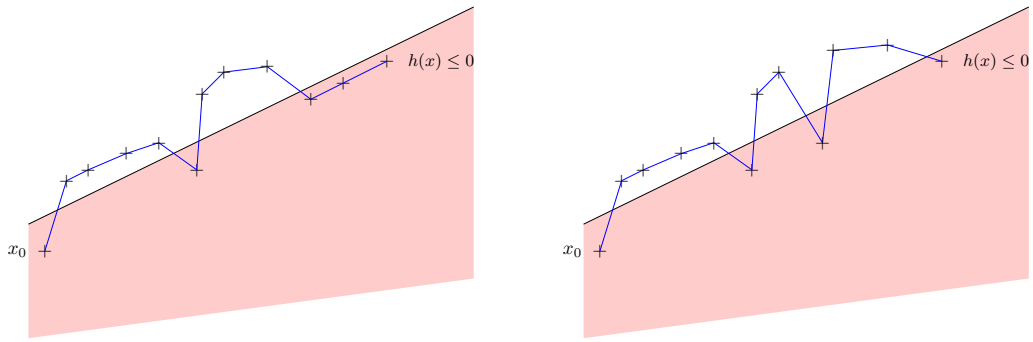

Figure 5.4: Left: Illustration of a trajectory initiated in x_0 which strongly p-satisfy the constraint $h(x) \leq 0$ for $p = 5$), Right:Example of a trajectory initialized in x_0 which weakly p-satisfy the constraint $h(x) \leq 0$ but is not strongly satisfying the same constrain with the same index.

5.2 p-Invariance for constrained reference tracking

5.2.1 p-invariance for controlled systems

This section presents an alternative approach to the traditional constrained tracking design methods which are based on a reference-governor type of mechanism coupled with a MPC technique for constrained handling [Bemporad, 1998]. Essentially, we aim to develop an attractive framework from the computational point of view that:

- guarantees recursive feasibility for a pre-defined region in the state space;
- builds on simplified constraints in the on-line optimization.

The recursive feasibility of constrained tracking can be guaranteed by the characterization of the *maximal controllable sets* [Blanchini and Miani, 2000]. Any initial state within this set can generate feasible trajectories by exploiting the controlled invariance and subsequently optimized with respect to a tracking criterion ([Blanchini and Miani, 2000]). We note however, that the characterization of the *maximal controllable set* is a notorious complex problem both in terms of off-line effort and complexity of the representation, which subsequently affects the on-line computational effort. As an alternative to the explicit use of the maximal controllable set, model predictive control (MPC) has been widely used with its receding horizon formulation. Recursive feasibility in MPC is related to the existence of an invariant terminal set. In the tracking case, this terminal set is

parameterized by a virtual feasible trajectory [Olaru and Dumur, 2005, Limon and Alamo, 2013, Falugi, 2015, Chisci and Zappa, 2003]. This parametrization leads to high on-line computational effort and has been the subject of research in different studies.

Reducing the complexity of the maximal controllable set or the terminal constraints in MPC by means of approximations can compromise the invariance property and consequently the recursive feasibility. This property preserves the recursive feasibility of optimization-based tracking control by maintaining a low computational effort.

We recall here the linear discrete-time model :

$$\begin{aligned} & x(k+1) = Ax(k) + Bu(k), \\ \text{subject to: } & \begin{cases} x(k) \in \mathcal{X}, & \forall\, k, \\ u(k) \in \mathcal{U}, & \forall\, k \end{cases} \end{aligned} \tag{5.21}$$

Practically, the controlled invariant set is chosen to be as large as possible in $\mathcal{X}$ to avoid conservativeness. The maximal controllable set can be approached off-line with an iterative procedure which implies projections of polyhedrons that can lead to a complex representation ([Nguyen et al., 2011a]). The objective in the remaining of the section is to introduce optimization-based tracking formulations building on a generalised invariance property developped in the previous subsection with the goal to circumvent the off-line and on-line complexity of the maximal controllable set.

Definition 5.10: Strong controlled p-invariance

A set $\mathcal{B} \subset \mathcal{X}$ containing the origin is said to be *strongly p-invariant* with respect to the constrained system (5.21) if there exists $p \in \mathbb{N}^*$ such that for all state $x(k) \in \mathcal{B}$, there exists one of the following options:

- a control action $u(k)$ such that $x(k+1) \in \mathcal{B}$
- a control sequence $(u(k), ..., u(k+p-1)) \in \mathcal{U}^p$ such that $x(k+p) \in \mathcal{B}$ and $(x(k+1), ..., x(k+p-1)) \in \mathcal{X}$.

The notion of p-invariance introduced here assumes the same periodicity index for any point in the set which engenders a trajectory which leaves the set. We show next that this characteristics can be relaxed to a certain extent and the tracking control associated carry on with simple modifications.

Definition 5.11: Weak controlled p-invariance

Given a compact convex set $\mathcal{B} \subset \mathbb{R}^n$ and $p \in \mathbb{N}^*$, $\mathcal{B}$ is said to be *weakly p-invariant* with respect to the system (5.31) if for any state $x(k) \in \mathcal{B}$ there exists $r \leq p$ and a control sequence $(u(k), ..., u(k+r-1)) \in \mathcal{U}^r$ such that $x_{k+r} \in \mathcal{B}$ and $(x(k+1), ..., x(k+r-1)) \in \mathcal{X}$.

In other words, any state in $\mathcal{B}$ returns into $\mathcal{B}$ in *at most* p number of steps.

Remark. *In the following, for the sake of brevity, the controlled p-invariance will be denoted p-invariance and the disambiguate with respect to the autonomous case is done by the dynamics under study.*

5.2.2 Strong p-invariance

The concept of strong p-invariance can be relevant in the control design for systems in the presence of constraints and relaxes the tracking objective whenever maximal controllable set is replaced by a simpler approximation thanks to the following theorem.

Theorem 5.9

Let the set Ω containing the origin be a controlled invariant set with respect to (5.21) and $C_N(\Omega)$ be the N-step controllable set to Ω for the same dynamics. Given a set $\mathcal{B}$ such that $\Omega \subseteq \mathcal{B} \subset C_N$. There exists an integer $p \in \mathbb{N}^*$ such that $\mathcal{B}$ is strongly p-invariant.

Proof

$\mathcal{B} \subset C_N(\Omega)$, so for any initial state $x(k) \in \mathcal{B}$ there exists a control sequence $(u(k), ..., u(k+N-1)) \in \mathcal{U}^N$ such that $x(k+N) \in \Omega \subset \mathcal{B}$. By fixing the index p to the maximal number of time steps N to reach Ω from $\mathcal{B}$ the existence is proved.

In other words, a simpler inner approximation of the maximal controllable set, while contains the attractive controlled invariant set, is necessarily strongly p-invariant.

5.2.3 Practical Construction of Strong p-Invariance

Given $\mathcal{B} \subset \mathcal{X}$ a bounded convex polyhedron containing the origin in its interior with $V = \{v_1, ..., v_{N_v}\}$ its vertices and their cardinality $N_v \in \mathbb{N}$. The strong

p-invariance of the set $\mathcal{B}$ with respect to (5.31) can be computed by Algorithm 2 below, which considers all the vertices of the candidate set $\mathcal{B}$ and tests the minimal contraction factor that can be obtained jointly along a time window of length p_s. The search for this contraction factor leads practically to a simple Linear Programming (LP) problem.

Under the assumption that the candidate set is contained in a controllable set $\mathcal{B} \subset \mathcal{C}_N(\Omega)$, the procedures ends in finite time. If this assumption does not hold, the condition $\lambda_s \mathcal{B} \supset \mathcal{X}$ guarantees that the algorithm will terminate in a finite number of steps. In this framework, it is important to observe that an explicit description of a controllable set $\mathcal{C}_N(\Omega)$ is not necessary in the above construction.

Algorithm 2: Strong p-Invariance

Input : The pair (A, B), the sets $\mathcal{X}$,$\mathcal{U}$ and $\mathcal{B}$
Output: The index p

$p_s = 0$, $\lambda_s = 1$
while $\lambda_s \geq 1$ **do**
  $p_s = p_s + 1$
  Solve:

$$\begin{aligned} \underset{\lambda_s,\, u_i}{\text{minimize}} \quad & \lambda_s \\ \text{subject to} \quad & A^{k-1} v_i + \sum_{j=0}^{k-2} A^j B u_{i,k-2-j} \in \mathcal{X}, \quad \forall k \in [2, p_s], i \in [1, N_v], \\ & A^{p_s} v_i + \sum_{j=0}^{p_s-1} A^j B u_{i,p_s-1-j} \in \lambda_s \mathcal{B}, i \in [1, N_v], \\ & (u_{v_i})_{i \in [1, N_v]} \in \mathcal{U}^{p_s} \end{aligned}$$

end

An alternative way to compute the index p exploits the equivalent formulation of the Theorem 5.8 for the controlled systems:

Theorem 5.10

A set $\Omega \subset \mathbb{R}^n$ containing the origin in its interior is strongly p-invariant with respect to the system (5.1) if and only if

$$\Omega \subset A^{-p}\Omega \bigoplus_{i=0}^{p-1} (A^{-i} B U) \cup A^{-1}\Omega \oplus (-BU) \tag{5.22}$$

The Algorithm 3 uses this set-based formulation to provide the strong index p. While A is Schur, A^{-1} is an expansive operator and then applied in the recursive

Algorithm 3: Strong p-Invariance (Alternative)

Input : The pair (A, B), the sets $\mathcal{U}$ and $\mathcal{B}$
Output: The index p

```
i=1
R_1 = A^{-1}B ⊕ (−BU)
R_i = R_1
while B ⊄ R_1 ⋃ R_i do
    R_i ← A^{-1}R_i ⊕ (−BU)
    i=i+1
end
p=i
```

scheme $R_{i+1} = A^{-1}R_i \oplus (-B\mathcal{U})$ the sets R_i tend to grow until the covering of the initial set.

5.2.4 Strong p-Invariance Based Reference Tracking

Based on the construction of strong p-invariant sets, let us now consider that the tracking control problem and exploit the existence of a p-invariant set $\mathcal{B}$ satisfying $\mathcal{B} \subset \mathcal{C} \subset \mathcal{X}$.

A prototype receding-horizon optimization for reference-tracking, which employs the p-invariance for a linear prediction model, will be denoted $\mathcal{O}(N_h, p, x(k))$ and formulated as follows :

$$
\begin{aligned}
J_p(x(k)) = \underset{(u(k),\ldots,u(k+M-1))}{\text{minimize}} \quad & \sum_{i=1}^{M} \|x^{ref}(k+i) - x(k+i)\|_Q^2 \\
\text{subject to} \quad & x(k+1+i) = Ax(k+i) + Bu(k+i)\ \forall i \in \{1,\ldots,M\}, \\
& x(k+i) \in \mathcal{X}\ \forall i \in [1,\ldots,M], \\
& u(k+i) \in \mathcal{U}\ \forall i \in [1,\ldots,M], \\
& x(k+p) \in \mathcal{B}
\end{aligned}
\tag{5.23}
$$

with $M = \max(N_h, p)$.

Proposition 5.4

$\mathcal{O}(N_h, p, x(k))$ is feasible for all $x(k) \in \mathcal{B}$.

Proof

For any state $x(k) \in \mathcal{B}$, there exists a control sequence $(u(k), ..., u(k+p-1)) \in \mathcal{U}^p$ such that $(x(k+1), ..., x(k+p-1)) \in \mathcal{X}$ and $x(k+p) \in \mathcal{B}$. Thus, in the case of a prediction horizon with $N_h \leq p$, the problem is feasible.
If $N_h > p$, the argument is slightly more elaborated and needs to rely on the invariance property of the controlled invariant superset $\mathcal{C}$. Indeed by construction $\mathcal{B} \subset \mathcal{C} \subset \mathcal{X}$, which implies the existence of a sequence $(u(k), ..., u(k+N_h-1)) \in \mathcal{U}^{N_h}$ such that $(x(k+1), ..., x(k+N_h-1)) \in C \subset \mathcal{X}$. □

Despite the result stated in Proposition 5.4, one cannot guarantee the recursive feasibility of the control strategy that implements the first part of the optimum control argument. This is because the *one-step invariance* property on $\mathcal{B}$ is not certified, and thus the feasibility of the optimization (5.23) on $\mathcal{B}$ does not imply the feasibility at iteration $k+1, \ldots$ as long as $x(k+p+1) \in \mathcal{B}$ does not hold.

To overcome the absence of recursive feasibility, a simple procedure can be constructed to enhance the p-invariance property. If $\mathcal{O}(N_h, p, x(k))$ designates the optimization (5.23), the main idea is to monitor the result of this optimization and to switch to a *safe return strategy* within the set $\mathcal{B}$ whenever the closed-loop trajectory leaves $\mathcal{B}$.

In order to simplify the switching criterion, the cost functions $J_i(x(k))$ and $J_p(x(k))$ will be compared for any $x(k) \in \mathcal{B}$. As long as the first one is less costly the procedure apply the first component of its control law. The following proposition provides a criterion that separates the case where the system remains in $\mathcal{B}$ from the case it leaves $\mathcal{B}$.

Proposition 5.5

Given the optimized cost $J_1(x(k))$ (resp $J_p(x(k))$) of optimization $\mathcal{O}(N_h, 1, x(k))$ (resp $\mathcal{O}(N_h, p, x(k))$). If $J_p(x(k)) < J_1(x(k))$ then $x(k+1) \notin \mathcal{B}$.

Proof

If U_p^* is the optimal solution of $\mathcal{O}(N_h, p, x(k))$, assume $x(k+1) \in \mathcal{B}$, then, every constraints of $\mathcal{O}(N_h, 1, x(k))$ are satisfied, U_p^* is a feasible solution of $\mathcal{O}(N_h, 1, x(k))$. Thus, the optimization of $\mathcal{O}(N_h, 1, x(k))$ provides a better solution, and $J_1(x(k)) \leq J_p(x(k))$.

Building on this preliminary result, the description of the procedure using strong p-invariance for tracking with recursive feasibility properties is presented in

Algorithm 4.

Algorithm 4: Strong p-Invariance based Reference Tracking

Input : $x_0 \in \mathcal{B} \subset \mathcal{X}$, N_{simu}, N_h, (A, B), $\mathcal{X}$,$\mathcal{U}$, $\mathcal{B}$
Output: $(x(k))_{k=0,...,N_{simu}}$, $(u(k))_{k=0,...,N_{simu}-1}$

$k = 1$, $i = 1$
while $k < N_{Simu}$ **do**
  if $i = 1$ **then**
    Solve $\mathcal{O}(N_h, 1, x(k)) \rightarrow (J_1^*, (u^1(k), ..., u^1(k + N_h - 1))^*)$
    Solve $\mathcal{O}(N_h, p, x(k)) \rightarrow (J_p^*, (u^1(k), ..., u^1(k + N_h - 1))^*)$
    if $J_1^* < J_p^*$ **then**
      $x(k + 1) = Ax(k) + Bu^1(k)$
    else
      $x(k + 1) = Ax(k) + Bu^p(k)$
      $i = p$
    end
  else
    Solve $\mathcal{O}(N_h, i, x(k)) \leftarrow (J_i^*, (u^i(k), ..., u^i(k + N_h - 1))^*)$
    if $J_1 < J_i$ **then**
      $x(k + 1) = Ax(k) + Bu^1(k)$
      $i = 1$
    else
      $x(k + 1) = Ax(k) + Bu^i(k)$
      $i = i - 1$
    end
  end
  $k = k + 1$
end

Proposition 5.6

The control law resulting from the recursive implementation of the first input of the optimal control sequence according to Algorithm 4 is recursively feasible for any initial state $x_0 \in \mathcal{B}$.

Proof

Assume the procedure is feasible at step k. Then the current state becomes $x(k + 1) \in \mathcal{X}$ and two cases have to be considered :

- If $x(k+1) \in \mathcal{B}$, then $\mathcal{O}(N_h, p, x(k+1))$ is feasible thanks to Proposition 5.4.
- If $x(k+1) \notin \mathcal{B}$, then $x(k+1)$ is part of a state sequence that began in $\mathcal{B}$. There exists an integer $q \leq p-1$ such that the state $x(k+1-q) \in \mathcal{B}$. So there exists a control sequence $(u(k+1-q), ..., u(k+1), ..., u(k-q+p-1)) \in \mathcal{U}^p$ such that $x(k+1-q+p) \in \mathcal{B}$ and $(x(k+2-q), ..., x(k-q+p)) \in \mathcal{X}^{p-1}$. Ignoring the tail, we conclude on the existence of a control sequence $(u(k+1), ..., u(k+1-q+p)) \in \mathcal{U}^{p-q}$ such that $x(k+2-q+p) \in \mathcal{B}$. This concludes the proof as long as $\mathcal{O}(N_h, p-q, x(k+1))$ is feasible.

5.2.5 Weak p-invariance for tracking

The next result formalize the relationship between *strong* and *weak* version of the p-invariance.

Theorem 5.11

If $\mathcal{B}$ is strongly controlled p-invariant, then $\mathcal{B}$ is weakly controlled p-invariant. Alternatively, if p_s (resp p_w) denotes the result of the computation of strong p-invariance (resp weak p-invariance), then $p_w \leq p_s$.

Proof

The proof is similar to the one for autonomous case by observing that in the definition of weak controlled invariance $q = p$ is a feasible choice for the number of steps for the return sequence.

Construction of Weak p-invariance: Given a bounded polyhedron $\mathcal{B} \subset \mathcal{X}$ with its set of vertices $V = \{v_1, ..., v_{N_v}\}$ and cardinality $N_v \in \mathbb{N}$, the weak p-invariance index can be computed by algorithm 5.

An alternative manner to compute the weak index p is to check the inclusion of $\mathcal{B}$ within the union of its preimages as formulated in the Theorem 5.8 for the autonomous systems and adapted here:

Algorithm 5: Weak p-invariance

Input : The pair (A, B), the sets $\mathcal{X}$, $\mathcal{U}$ and $\mathcal{B}$
Output: the index p

for $v_i \in V$ **do**
  $p_w^i = 0$; $\lambda_w = 1$
  while $\lambda_w \geq 1$ **do**
    $p_w^i = p_w^i + 1$
    Solve:

$$\begin{aligned} \underset{\lambda_w,\,(u_i)_{i\in\{1,p_w^i-1\}}}{\text{minimize}} \quad & \lambda_w \\ \text{subject to} \quad & A^{k-1}v_i + \sum_{j=0}^{k-2} A^j B u_{k-2-j} \in \mathcal{X},\ \forall k \in \{2, p_w^i\}, \\ & A^{p_w^i}v_i + \sum_{j=0}^{p_w^i-1} A^j B u_{p_w^i-1-j} \in \lambda_w \mathcal{B},\ i \in \{1, p_w^i - 1\} \end{aligned}$$

  end
end
$p_w = \max(p_w^1, ..., p_w^{N_v})$

Theorem 5.12

A set $\Omega \subset \mathbb{R}^n$ containing the origin in its interior is weakly p-invariant with respect to the system (5.1) if and only if

$$\Omega \subset \bigcup_{i=1}^{p} A^{-i}\Omega \oplus (-B\mathcal{U}) \tag{5.24}$$

The Algorithm 6 uses this set-based formulation to provide the weak index p.

Proposition 5.7

Given a convex compact polyhedron $\mathcal{B}$ containing the origin, and $\{p_1, ..., p_{N_v}\}$ the p-indices of vertices $\{v_1, ..., v_{N_v}\}$ computed based on algorithm 5 is thus leading to a weak p-invariance index for the entire set. Then, for any state

Algorithm 6: Weak p-Invariance (Alternative)

Input : The pair (A, B), the sets $\mathcal{U}$ and $\mathcal{B}$
Output: The index p

```
i=1
R_1 = A^{-1}B ⊕ (−BU)
R_i = R_1
while B ⊄ R_1 ⋃ R_2 ⋃ ··· ⋃ R_i do
    R_i ← A^{-1}R_i ⊕ (−BU)
    i=i+1
end
p=i
```

$x(k) \in \mathcal{B}$,one can use the same periodicity as its vertex component with highest periodicity.

A tracking procedure using weak p-invariance and based on the optimization (5.23) can be proposed as in algorithm 7. The principle behind the properties of recursive feasibility of the tracking procedure is summarized next, the proof follows the arguments in Section 5.2.2:

- If $x(k) \in \mathcal{B}$, the cost functions for an optimization problem with 1-step invariance constraints and p-step invariance constraints are compared.

- If the system leaves $\mathcal{B}$, then every costs from 1-step to p-steps return to $\mathcal{B}$ are compared. In other words, the validation of the weak p-invariance is tested at each iteration in order to find the shortest return path into $\mathcal{B}$.

5.2.6 Tracking IBC with p-invariance

IBT is the main topic of the Chapter 4, we consequently refer to the same formulations and notations. The IBT design is built on two inner and outer convex compact controlled-invariant sets containing the origin Ω^o and Ω^v, where $\Omega^o \subset \Omega^v \subset \mathcal{X}$. These inner and outer sets have to be re-scaled and translated in order to contain the origin of the dynamical system governing the tracking error. The p-invariance property is preserved by homogeneous transformations.

Given the outer set Ω^v and assume its strong p-invariance, the virtual reference

Algorithm 7: Weak p-Invariance Reference Tracking

Input : $x_0 \in \mathcal{B} \subset \mathcal{X}$, N_{simu}, N_h, (A, B), $\mathcal{X}$, $\mathcal{U}$, $\mathcal{B}$
Output: $(x(k))_{k=0,...,N_{simu}}$,$(u(k))_{k=0,...,N_{simu}-1}$ $k = 1$, $i = 1$

while $k < N_{simu}$ **do**
  if $i = 1$ **then**
    Solve $\mathcal{O}(N_h, 1, x(k)) \rightarrow (J_1^*, (u^1(k), ..., u^1(k + N_h - 1))^*)$
    Solve $\mathcal{O}(N_h, p, x(k)) \rightarrow (J_p^*, (u^p(k), ..., u^p(k + N_h - 1))^*)$
    if $J_1^* < J_p^*$ **then**
      $x(k + 1) = Ax(k) + Bu^1(k)$
    else
      $x(k + 1) = Ax(k) + Bu^p(k)$ $i = p$
    end
  else
    Solve $\mathcal{O}(N_h, i, x(k)) \rightarrow (J_i^*, (u^i(k), ..., u^i(k + N_h - 1))^*)$
    Solve $\mathcal{O}(N_h, i - 1, x(k)) \rightarrow (J_{i-1}^*, (u^{i-1}(k), ..., u^{i-1}(k + N_h - 1))^*)$
      $\vdots$
    Solve $\mathcal{O}(N_h, 1, x(k)) \rightarrow (J_1^*, (u^1(k), ..., u^1(k + N_h - 1))^*)$
    $q^* = \arg \min_{q \in \{1,i\}} (J_q)$
    $x(k + 1) = Ax(k) + Bu^{q^*}(k)$
    $i = i - 1$
  end
end

and the scaling factor are solutions of the optimization problem $\mathcal{R}(x^{ref}(k), x(k))$:

$$\begin{aligned} \underset{(\bar{x}(k), \bar{u}(k), \alpha(k))}{\text{minimize}} \quad & \|x^{ref}(k) - \bar{x}(k)\|^2 \\ \text{subject to} \quad & \bar{x}(k) = A\bar{x}(k) + B\bar{u}(k), , \\ & \bar{u}(k) \in (1 - \alpha(k))\mathcal{U}, \\ & x(k) \in \{\bar{x}(k)\} \oplus \alpha(k)\Omega^v \subset \Omega^v. \end{aligned} \tag{5.25}$$

The main idea is to solve, for the current state $x(k)$ in Ω^v, the optimization problem $\mathcal{R}(x^{ref}(k), x(k))$ and once the admissible $\bar{x}(k)$ and $\bar{u}(k)$ are computed to regulate the tracking error by solving a IBC problem. This tracking error will be defined as $\epsilon(k) = \bar{x}(k) - x(k) \in \alpha(k)\Omega^v$ and two cases have to be considered:

- if $\epsilon(k + 1) \notin \Omega^v$ as a result of the IBC at step k then we hold $\bar{x}(k)$ and $\bar{u}(k)$ for maximum p steps in order to allow $\epsilon(k + i) \in \Omega^v$ for some $0 < i \leq p$. Practically, at each iteration i, we check if $\epsilon(k+i) \in \mathcal{B}$ and if this is the case we release $\bar{x}(k + i)$ and $\bar{u}(k + i)$

- if $\epsilon(k+1) \in \Omega^v$ as a result of the IBC at step k we start from the beginning the procedure with virtual reference design and IBC.

Practically, the optimization (5.25) generates a virtual trajectory which are fixed points of the system (5.31). Whenever the reference to be tracked $x^{ref}(k)$ converges to a fixed point, it can be shown that $\bar{x}(k)$ converges to a feasible fixed point $\bar{x} \in \mathcal{X}$ with respect to an admissible control $\bar{u} \in \mathcal{U}$ and the the tracking error $\epsilon(k)$ asymptotically decreases to zero using the IBC. The formal proofs are adapting the IBC properties for the case of p-invariance similar to the previous chapter and are not presented for brevity.

5.2.7 Illustrative numerical examples

Example: p-invariance for tracking

Consider the linear double integrator :

$$x(k+1) = \begin{bmatrix} 1 & 0.08 \\ 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} 0.0032 \\ 0.08 \end{bmatrix} u(k) \tag{5.26}$$

subject to constraints:

$$-2.6 \leq x_1(k) \leq 2.6, -3 \leq x_2(k) \leq 3, -5 \leq u(k) \leq 5. \tag{5.27}$$

The two techniques presented in the section are considered for simulation and comparison: strong and weak p-invariant reference tracking and p-IBC. Then two types of trajectories are presented:

- A dynamically generated trajectory with the goal to test and illustrate the recursive feasibility and the reactivity of both tracking algorithms;
- A sequence of switching of fixed points which aims to test (aside the recursive feasibility) the convergence properties.

The inner polyhedral set Ω^o considered is the maximal admissible set with respect to a linear feedback law:

$$u(k) = -\begin{bmatrix} 16.65 & 6.23 \end{bmatrix} x(k) \tag{5.28}$$

A controlled invariant approximation of the maximal controllable set $\mathcal{C}(\Omega^o)$ is also computed as a N-step controllable set,
The candidate p-invariant set is represented by a simple inner approximation of $C_N(\Omega^o)$, represented in blue in Fig.5.5. The same set, denoted Ω^v, is used as an outer set for the p-invariant MPC and IBT strategies.

Time-Varying (Dynamic) Trajectory

In the first simulation scenario, the reference is generated using the dynamical model (5.26) based on a excitation signal which violates drastically the imposed constraints. As a result, the time-varying reference trajectory leaves the state constraints set as illustrated by dashed trajectories in Fig.5.5.

The initial state of the system is selected on the frontier of Ω^v on a extreme state (corresponding to zero speed if the state is interpreted in terms of position-speed coordinates). For the comparative study, strong and weak p-invariant sets have been constructed and the p index has been computed using Algorithms 2 and 5 to be $p = 14$. Using this low complexity set (4 vertices) the reference tracking optimization has been solved according to the Algorithms 4 and 7 and the results confirm the recursive feasibility of both receding-horizon reference-tracking algorithms.
Even if globally the behaviour is similar, a slight difference can be observed when the state leaves $\mathcal{B}$ (the signals are depicted with respect to the time in Figure 5.6).

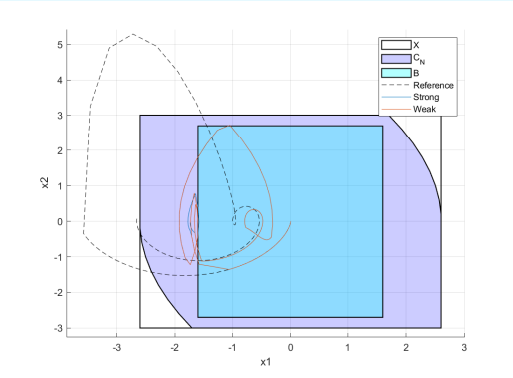

Figure 5.5: Trajectories in the state space of reference (dashed), Strong p-invariance procedure (blue) and Weak p-invariance procedure (red).

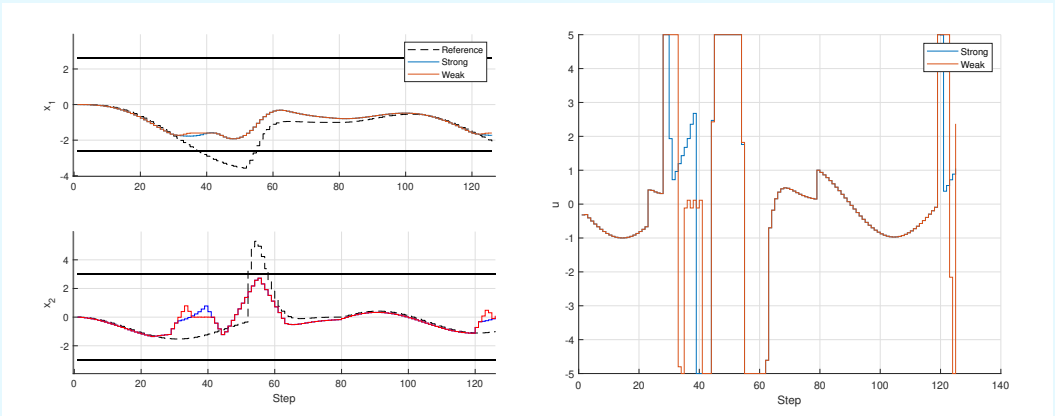

Figure 5.6: Temporal trajectories of reference (dashed), Strong p-invariance (blue) and Weak p-invariance (red). Right: Control action of Strong p-invariance (blue) and Weak p-invariance (red).

Static Reference Trajectory

In this case, the reference is a fixed-point out of the admissible set and this reference commutes to a symmetric fixed point out of the admissible set at a regular time interval. The invariant set Ω^v is the strongly p-invariant set with index $p = 8$. Figures 5.7 and 5.9, compare strong and weak p-invariant based reference tracking which provide a recursive feasible control law and good performances.

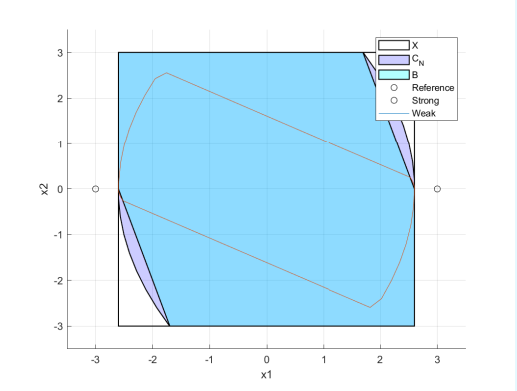

Figure 5.7: Trajectories in the state space of reference (dashed), Strong p-invariance (blue) & Weak p-invariance (red).

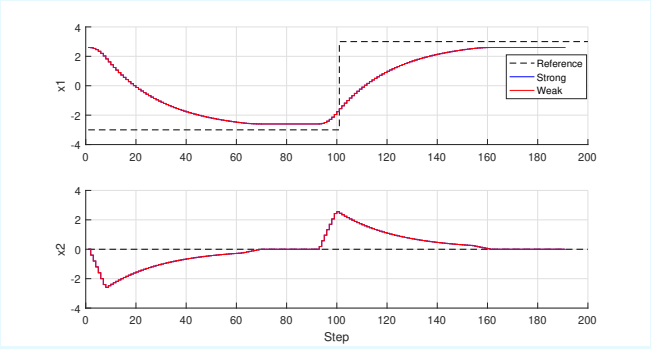

Figure 5.8: Trajectories of reference (dashed) and states trajectory strong p-invariance (blue) and weak p-invariance(red)

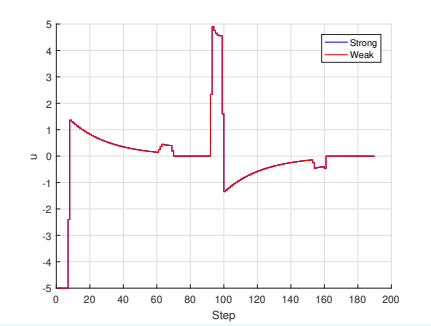

Figure 5.9: Control actions: strong p-invariance (blue) and weak p-invariance(red)

In order to complete the design with stability (convergence) guarantees for such piecewise constant references, an IBC for tracking is implemented using the periodic invariance notion. The closed-loop performance is depicted in Fig.5.10, 5.12. It is important to observe that although the virtual reference is a fixed point, it is updated at each iteration, and thus leads to a sequence of feasible set-points for the IBC procedure.

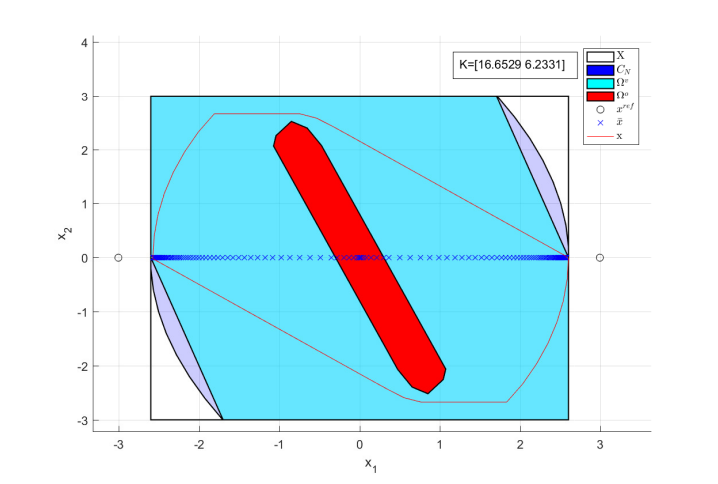

Figure 5.10: Trajectories in the state space of reference (dashed), virtual $\bar{x}$ (blue cross) and state trajectory (red)

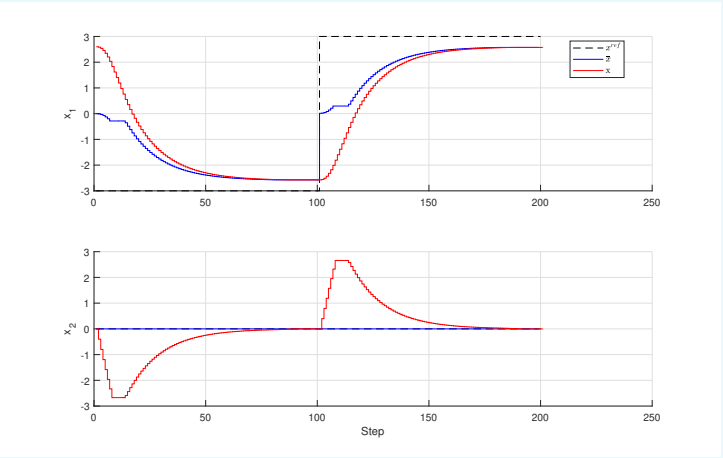

Figure 5.11: Time trajectories of reference (dashed), virtual trajectory $\bar{x}$ (blue) and and state trajectory (red)

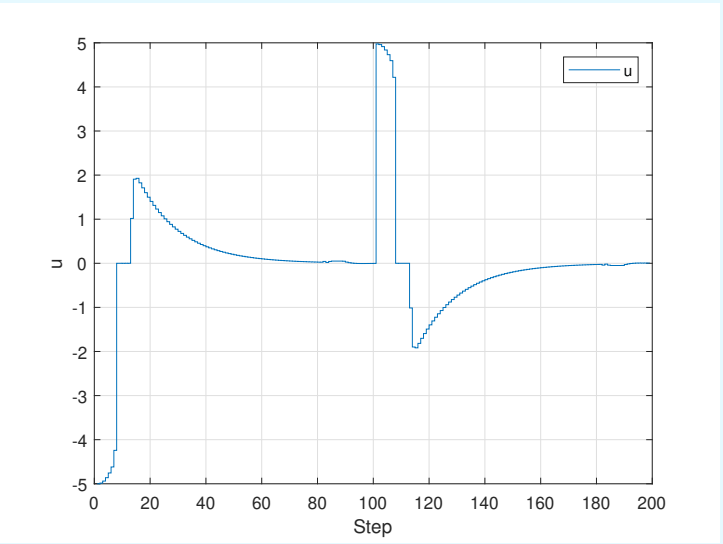

Figure 5.12: Control action of p-invariant IBT

5.3 MCA based on p-invariant sets

5.3.1 Model of the dynamics

In this section we focus on a real-time implementation of MCA strategies based on strong p-invariance. We recall here the rail dynamic models according to the two cases presented on Section 2.2.2:

1. The computationally attractive one with $x(k) = \begin{bmatrix} p(k) & v(k) \end{bmatrix}^T$ and $u(k) = a(k)$:

$$x(k+1) = \begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} \frac{1}{2}T_s^2 \\ T_s \end{bmatrix} u(k) \tag{5.29}$$

2. The model using the jerk as an input with $x(k) = \begin{bmatrix} p(k) & v(k) & a(k) \end{bmatrix}^T$ and $u(k) = j(k)$:

$$\begin{cases} x(k+1) &= \begin{bmatrix} 1 & T_s & \frac{1}{2}T_s^2 \\ 0 & 1 & T_s \\ 0 & 0 & 1 \end{bmatrix} x(k) + B = \begin{bmatrix} \frac{1}{6}T_s^3 \\ \frac{1}{2}T_s^2 \\ T_s \end{bmatrix} u(k) \\ y(k) &= \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} x(k) \end{cases} \tag{5.30}$$

That we formalize in the following:

$$\begin{cases} x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= Cx(k) \end{cases} \tag{5.31}$$

5.3.2 A MPC approach

In this subsection we aim to exploit the alleviated MPC framework presented in Section3.2.2 but using Maximal controllable set approximations and strong p-invariance. In order to exemplify the concept let us consider two sets $\widetilde{C}_N^1$ and $\widetilde{C}_N^2$ for models (5.30) and (5.29) that are respectively p_1-invariant and p_2-invariant with $p_1 = 15$ and $p_2 = 19$, these sets are depicted on Fig.5.13.

Considering controlled strongly p-invariant sets, several finite-time optimization problems denoted by $\mathcal{P}^i(p, x(k))$ can be rewritten for the same model i but with a different optimization cost and optimization structure ($i \in \{1, 2\}$ referring to p_1 and p_2):

$$\begin{aligned}
\underset{(u(k),\ldots,u(k+p-1))}{\text{minimize}} \quad & J^i(k)(x(k+1),\ldots,x(k+p),u(k),\ldots,u(k+p-1)) \\
\text{subject to} \quad & x(k+1+j) = Ax(k+j) + Bu(k+j), \quad \forall j \in \{1,p\}, \\
& x(k+j) \in \mathcal{X}, \quad \forall j \in \{1,p-1\}, \\
& u(k+j) \in \mathcal{U}, \quad \forall j \in \{0,p-1\}, \\
& x(k+p) \in \widetilde{C}_N
\end{aligned} \tag{5.32}$$

These optimization problems will replace the classic MPC formulation in order to decrease based on switching the complexity both off-line and on-line. Practically, complexity of $\mathcal{X}$ is low (a square for a 2D state space and a cube for a 3D state space) and $\widetilde{C}_N$ one's is of the same order, this formulation is consequently less complex than the alleviated predictive controller presented in (3.13).

Consideration of simpler p-invariant sets implies a loss of recursive feasibility from a classical MPC framework point of view, to compensate for this loss, Algorithm 8 is described next:

- If the current state is in $\widetilde{C}_N$, then the optimization problem (5.32) is solved for a horizon p.
- Else the state left $\widetilde{C}_N$, and the problem is solved with a decreasing horizon until the state returns in $\widetilde{C}_N$.

Algorithm 8: Tracking with p-invariant MPC

```
Initialization (x_0 ∈ B ⊂ X)
k = 1
j = 1 % index counting the steps of a periodic window
while k < N_Simu do
    if x(k) ∈ C̃_N then
        j=p
        Solve P^i(p, x(k)) → (u^1(k), ..., u^1(k+p-1)))*
    else
        j=j-1
        Solve P^i(j, x(k)) → (u^1(k), ..., u^1(k+j-1)))*
    end
    x(k+1) = Ax(k) + Bu^1(k)
    k = k+1
end
```

Proposition 5.8

The iterative procedure in Algorithm 8 is recursively feasible if the initial state is in $\widetilde{C}_N$.

Proof

Assume the procedure is feasible at step k.

- If $x(k) \in \widetilde{C}_N$ which is controlled p-invariant then it exists $(u(k), ..., u(k+p-1)) \in \mathcal{U}^p$ such that $(x(k+1), ..., x(k+p-1)) \in \mathcal{X}$ and $x(k+p) \in \widetilde{C}_N$.
- If $x(k) \notin \widetilde{C}_N$, there exists $j \in \{1, p-1\}$ such that $x_{k-j} \in \widetilde{C}_N$. Consequently, there exists a control sequence $(u(k-j), ..., u(k-j+p-1)) \in \mathcal{U}^p$ such that $(x(k-j+1), ..., x(k), ..., x(k-j+p-1)) \in \mathcal{X}$ and $x_(k-j+p) \in \widetilde{C}_N$. As a consequence of these two properties there always exists a control sequence that leads states toward $\widetilde{C}_N$. The receding horizon optimization is in charge of selecting the horizon p at each time step in order to enforce this property.

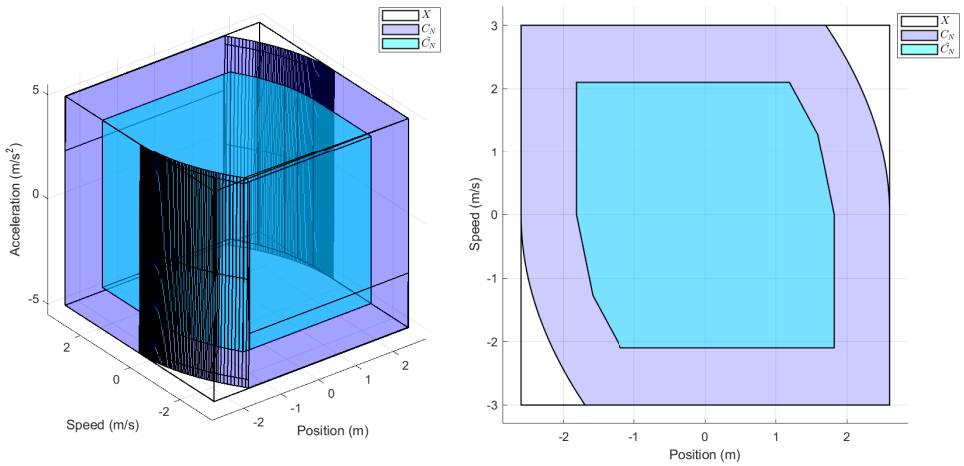

Figure 5.13: Left: 3D State space representation for model (5.30), C_N and $\widetilde{C_N}$, Right : 2D State space representation for model (5.29), C_N and $\widetilde{C_N}$

Example:

The study case elaborated in this simulation considers the acceleration reference associated to a signal perceived during a slalom. Here, results from simulations of MPC-MCA from (3.13) are compared with results from application of Algorithm 8 to models (5.30) and (5.29). The p-invariant set is obtained by contracting an inner approximation of the maximal controllable set with a periodic index p_1.The chosen prediction horizon is similar to invariance periods : $N_h = p_1$.
Weightings in the criteria are $q_y = 100$, $Q_x = diag(100, 1, 50)$ and $R = 1$ for model (5.30), $Q_x = diag(100, 1)$ and $R = 50$ for model (5.29).
Acceleration is rendered with timely reaction and respecting the profile although the constraints are activated. In particular the shape of the reference profile is respected (5.14) and it is to highlight that the periodic invariant MPC applied to model (5.29) reaches limitations in acceleration and is consequently closer to the reference. However responses of periodic invariant MCAs are affected by switching of the cost function which affects the acceleration. All three algorithms are conservative as they don't use the whole workspace in position (5.14) although the 2D system manages the workspace in a better way, positioning the virtual car closer to the center of symmetry.

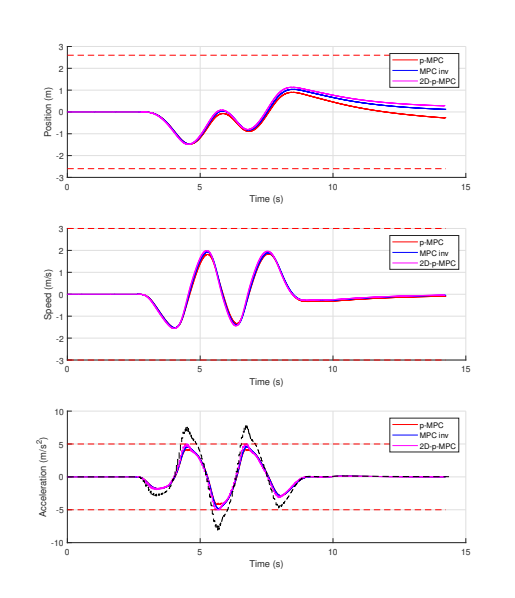

Figure 5.14: Top : Positions in function of time, Middle : Speeds in function of time, Bottom : Accelerations in function of time (reference signal dashed)

On Figure 5.15, the jerk is presented as a control action for alleviated MPC (3.13) and Algorithm 8 applied to model 5.30. We can observe a more aggressive behavior of control action when state trajectory leaves $\widetilde{C}_N^1$ as the system tends to return within this latter set.

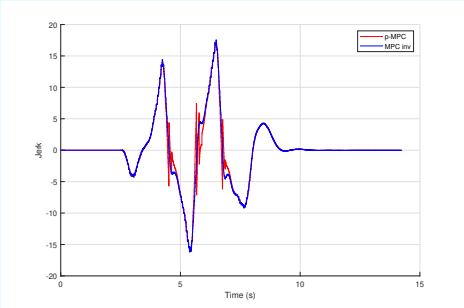

Figure 5.15: Jerk in function of time for model (5.30)

From a computational point of view, Table 5.1 shows the theoretical decrease of the number of constraints with respect to optimization problems to solve during procedures.
Figure 5.17 represents computation time of optimization problem at each iteration as a function of time and confirms the decrease of computational burden. In this situation, comparison between models (5.30) and (5.29) is not obvious because p_2 is slightly higher than p_1 on one hand and the state trajectory remains in $\widetilde{C}_N$ in the 2-dimensions case so the optimization is always performed in the worst case as it is shown on Figure 14 and on Figure 14.

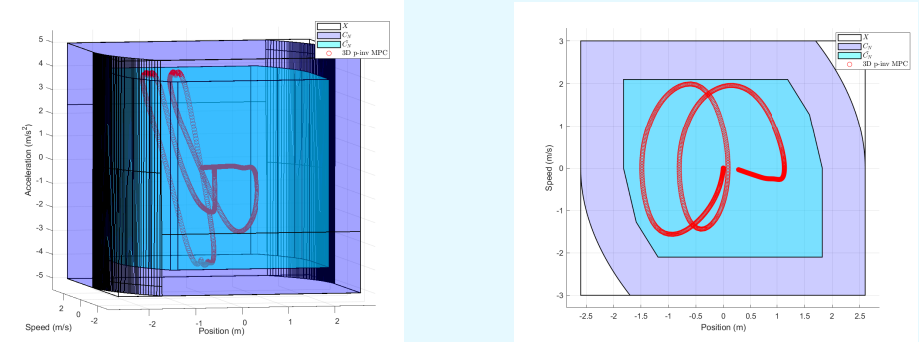

Figure 5.16: Left: 3D State space trajectory for p-invariant algorithm for model (5.30). Right: 2D State space trajectory for p-invariant algorithm for model (5.29)

Table 5.1: Complexity of Optimization problem

MPC Formulation	Number of constraints
Model (5.30) Invariant MPC	489
Model (5.30) p-invariant MPC	170 (worst case)
Model (5.29) p-invariant MPC	152 (worst case)

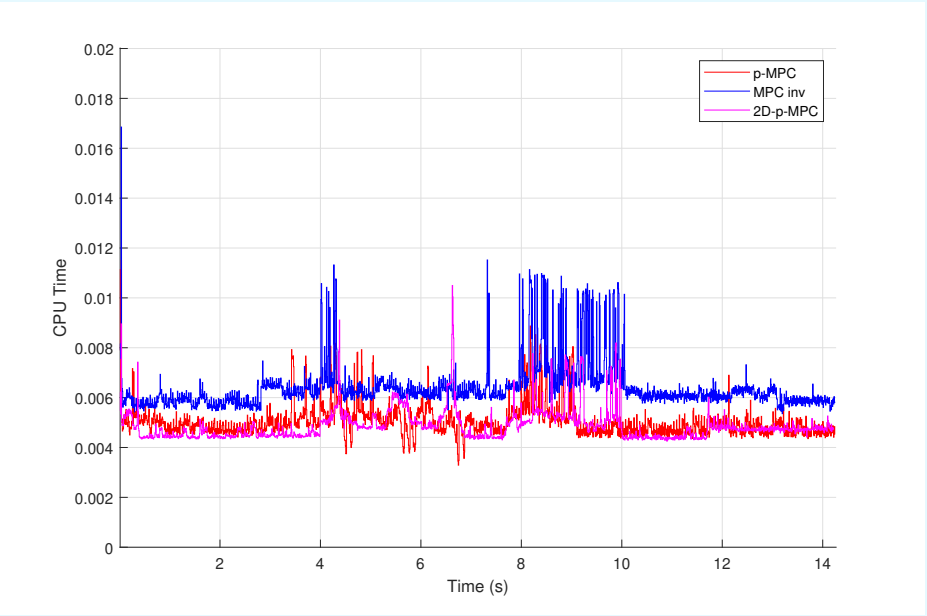

Figure 5.17: Execution time for the different strategies.

Chapter 6

Conclusion and perspectives

6.1 Main contributions emerging from the present work

In the Chapter 1 we defined the driving simulation framework by explaining how to stimulate inertial sensations by movement and rotation of a platform thanks to the tilt coordination technique (Figure 1.2) applied to an electromechanical structure made of rails and hexapod (Figure.1.3). We summarized the operating of those kind of dynamic driving simulators (Figure 1.4) and provided some elements of their history from early flight simulation to high performance driving simulation. We noticed that techniques used in flight simulation had been adapted to driving simulation particularly for the high frequency acceleration restitution.

The function that allow the conversion of a specific force to be rendered into position and tilt orders is commonly called "Motion Cueing Algorithm" or "MCA" (Figure 2.1) and is a topic of interest of many research works in the last decades. A part of those techniques were reviewed in the Chapter 2. Historically, MCA was conceived based on filters (Figure 2.4) and progressively enhanced with the optimization philosophy first with adaptative filters, then with Model Predictive Control (MPC). This latter technique is undoubtedly the most promising actually but has to cope with main limitations such as the driver's vestibular mechanics awareness, the computation time, the delays compensation or the prediction issues. The present thesis proposed theoretical notions and methodological solutions to address those issues.

The Chapter 3 first exposed the structure of the subsystems in order to take into account for the nonlinearities of gravitationnal field projections on the driver's

frame (Section 2.1.3). We proposed in-line predictive structure splitting the linear and rotational dynamics prioritising one and compensating the gap thanks to the other (Figures 3.3-3.8). The two strategies were compared in terms of performance and computation time and the results were discussed. Second, we proposed a lighter MCA design for delay compensation (Figure 3.15). The goal was to use information about maximal controllable set in the state space to adapt the state constraints (including the terminal one) into the set-inclusion of the only next predicted state with respect to the (approximation) of the maximal controllable set.

In the Chapter 4 we adapt the Interpolation Based Control (IBC) computationally promising properties to the tracking framework with the objective to use it in the MCA framework. This technique favours an optimal use of the state space and doesn't need a prediction horizon. We demonstrate its stability and recursive feasibility properties and show that it can fulfil this objective. The computational complexity is linked to the own complexity of the set.

The latter interpolation principle remains valid in a larger framework, consequently the main idea of the Chapter 5 is to simplify the set considered in optimization based control by exploiting the property of p-invariance. Thus we can build lighter inner approximations of controllable sets first for autonomous systems and then for controlled ones. Finally this novel technique was used for MCA application.

6.2 Perspectives

The main issues of MCA design can be summarized in by three features:

- **the rendering performance** which is linked to the minimization of motion sickness and conservativeness (the workspace shall be optimally used with respect to a performance index which is inherent to each of the drivers)
- **the tuning difficulty** *i.e* the number of parameters (weigths, prediction horizon) to set up. More parameters means more complexity and interdependence among them.
- **the computational burden** as a consequence of the real-time requirements and the interdependence in between the optimization-based modules.

From an engineer point of view, a desirable (if not ideal) MCA should combine simultaneously the three properties but it seems there exists a structural compatibility issue between them which is summarized in the Venn diagram 6.1: Practi-

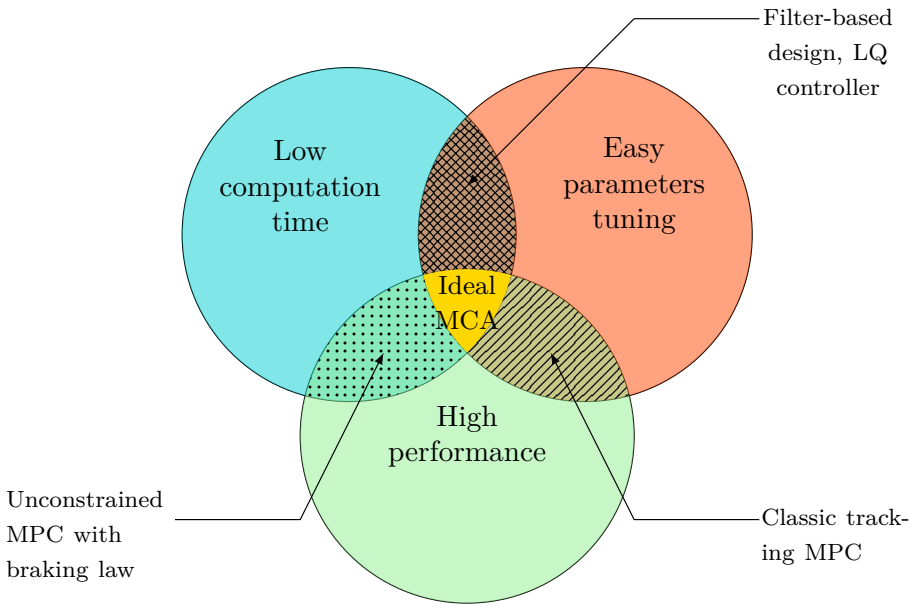

Figure 6.1: Venn diagram illustrating the structural issues to design an ideal MCA

cally we can recognize some situations validating two features but not the third such as:

- The filter-based MCA allows for example a simplicity of design and a low computation time but has poor performance.
- The MPC controller practically uses a solver for the constrained optimization which iteratively seaks the global minimum of a quadratic cost function within the set of constraints. This operation may be computationally heavy when the number of states or the prediction horizon increases and with the constraint activation. However, this type of controller is able to use the whole workspace and is easy to tune when the number of weights is low.
- A solution to decrease the computation time without degrading the performance is to use the braking law with an unconstrained MPC framework. Practically the unconstrained MPC optimization problem is solved by cancelling the gradient and when the platform is too close to the workspace limitations, the braking law is applied. However, the enhancement implies more parameters to tune.
- Also when the prediction horizon is low, the computational cost is better but the performance is worse except if we set up a pre-positioning and in this

case, it increases the number of parameters.

6.3 On a broader scope

The dynamic driving simulation is a complex domain involving very different fields (automatic control, mechanics, physiology, psychology and computer science,...) that interact with each other. For example, on one hand, the studies on the driver's behavior depends on the quality of the restitution of the sensations (the reactions should be the identical as the ones on a real drive) and on the other hand those behavioral studies are a vector of improvement for the restitution algorithms.

Also, the simulation platform control is tributary of the electro-mechanical design of the platform which is not necessarily build for a priori known restitution algorithms.

The subjective experience makes the precise sensations rendering too complicated from an engineering point of view. Assumptions and simplifications are essential knowing their impacts on the subject.

Consequently, the progress in the driving simulation fields will be pluri-disciplinary as well as experimental with clear objective criteria in order to easily compare the different strategies of MCA.

Appendix A

Mathematical tools

A.1 Mathematical tools for control

We recall in this section some fundamental tools from linear algebra and topology in order to set up in a synthetic way the convex optimization framework to finally introduce optimal control.

Assumption 1.1

In this section, n, m, p, l and s refer to positive integers.

A.1.1 Linear algebra and topology

The two next definitions introduce the notion of weighting of quadratic norms used in optimal control.

Definition A.1: Positive (semi)definite matrix

A matrix $Q \in \mathbb{R}^{n \times n}$ is said to be **positive definite** (respectively **semidefinite**) and denoted by $Q \succ 0$ (respectively $Q \succeq 0$) if:

1. Q is symmetric, *i.e* $Q^T = Q$
2. for all non-zero vector $x \in \mathbb{R}^n$, $x^T Q x > 0$ (respectively $x^T Q x \geq 0$) .

Definition A.2: Weighted quadratic norm

If $Q \succeq 0$ and $x \in \mathbb{R}^n$, then the **Q-weighted quadratic norm** of x denoted $\|x\|_Q$ is given by:

$$\|x\|_Q = \sqrt{x^T Q x} \tag{A.1}$$

If Q is a diagonal matrix ($Q = diag(q_1, \ldots, q_n),\ q_i \geq 0\ \forall i$), then the quadratic norm is derived from:

$$\|x\|_Q^2 = \sum_{i=1}^{n} q_i x_i^2 \tag{A.2}$$

which means each component has its proper weight in the norm. In the following we denote $\|.\|$ the quadratic norm weighted by the identity matrix (equal weights on vector's components.

Throughout the resolution of a linear differential system of the first order used to model the dynamics of linear system (see Definition.A.19) we can express the solution as a function of an exponential of a matrix whose definition is given below:

Definition A.3: Exponential of a matrix

Given a square matrix $A \in \mathbb{R}^{n \times n}$, the **exponential of the matrix** A, denoted by e^A is the matrix given by:

$$e^A = \sum_{n=0}^{\infty} \frac{A^n}{n!} = I_n + A + \frac{A^2}{2} + \ldots \tag{A.3}$$

We recall below three fundamental definitions of topology used in convex analysis.

Definition A.4: Open ball

An **open ball** of center $a \in \mathbb{R}^n$ of radius $r > 0$ is the set:

$$\mathcal{B}(a, r) = \{x \in \mathbb{R}^n \mid \|x - a\| < r\} \tag{A.4}$$

Definition A.5: Open and closed sets

Let $\mathcal{C} \subset \mathbb{R}^n$. $\mathcal{C}$ is said to be **open** if and only if:

$$\forall a \in \mathcal{C}\ \exists r > 0\ \mathcal{B}(a, r) \subset \mathcal{C}$$

$\mathcal{C}$ is said to be **closed** if and only if $\mathbb{R}^n \setminus \mathcal{C}$ is open

Definition A.6: (Affine) Hyperplane

A subset of $\mathbb{R}^n$ of dimension $n-1$ is called a **hyperplane**. An hyperplane $\mathcal{H}$ is said to be **affine** if there exists $a \in \mathbb{R}^n$ and $b \in \mathbb{R}$ such that:

$$\mathcal{H} = \left\{ x \in \mathbb{R}^n \mid a^T x = b \right\} \tag{A.5}$$

A.1.2 Convexity and Optimization

The three next definitions characterising convex sets and functions that are of main interest for optimization.

Convexity

Definition A.7: Convex Set

A set C is convex said to be convex if:

$$\forall (x, y) \in C^2 \quad \forall \alpha \in [0, 1] \quad \alpha x + (1 - \alpha) y \in C \tag{A.6}$$

In other words, a set is convex if for all couples of elements the segment linking them is included in the set.

Definition A.8: Convex function

Let $C \subset \mathbb{R}^n$ a convex set. A function $f : C \mapsto \mathbb{R}$ is said to be **convex** if:

$$\forall (x, y) \in C^2\ \forall t \in [0, 1]\ \ f(tx + (1-t)y) \leq t f(x) + (1-t) f(y) \tag{A.7}$$

equivalently, a f is convex if and only if its epigraph $\mathcal{E} = \{(x, \xi) \in C \times \mathbb{R} \mid f(x) \leq \xi\}$ is a convex set.

Definition A.9: Convex Hull

Given a set $C \subset \mathbb{R}^n$, the Convex hull of C, denoted by $Conv(C)$, is the smallest convex subset of $\mathbb{R}^n$ containing C.
Consequently, any element of $Conv(C)$ can be expressed as a finite convex combination of element of C.

$$x \in Conv(C) \Longleftrightarrow x = \sum_{i=1}^{p} \lambda_i x_i \tag{A.8}$$

where $p \in \mathbb{N}^*$, $(\lambda_i)_{i=1,\dots,p} \in [0,1]^p$ and $(x_i)_{i=1,\dots,p} \in C^p$ such that $\sum_{i=1}^{p} \lambda_i = 1$.

In optimal control theory the physical constraints are mainly modeled through half-spaces, when several constraints has to be hold at the same time, the resulting set of contraints is said to be polyhedral or polytopic while bounded and can have 2 distincts formalization which are depicted in the following:

Definition A.10: Polyhedron - Half-space Representation

A half-space is a set that can be defined as

$$\mathcal{H} = \left\{ x \in \mathbb{R}^n \mid a^T x \leq b \right\} \tag{A.9}$$

with $a \in \mathbb{R}^n$ and $b \in \mathbb{R}$.

A Polyhedron (or polyhedral set) $\mathcal{P}$ is the intersection of a finite number $m \in \mathbb{N}^*$ of half-spaces.

$$\mathcal{P} = \{ x \in \mathbb{R}^n \mid Fx \leq g \} \tag{A.10}$$

where $F \in \mathbb{R}^{m \times n}$ and $g \in \mathbb{R}^m$.

Example:Unbounded Polyhedron

Figure.A.1 depicts the unbounded polyhedron defined by the intersection of two halfspaces:

$$\mathcal{H}_1 : \ -x_1 + x_2 \leq 4 \text{ and } \mathcal{H}_2 : \ x_2 \leq 3 \tag{A.11}$$

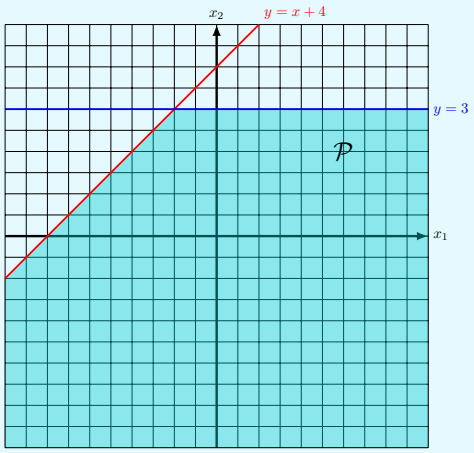

Figure A.1: Example of a unbounded Polyhedron

Definition A.11: Polytope

A polytope is a bounded polyhedron.

Definition A.12: Polytope - Vertex Representation

If $\mathcal{P}$ is a polytope, there exists a finite number of elements $(v_1, \ldots, v_{N_v})$ with $N_v \in \mathbb{N}$ such that:

$$\mathcal{P} = \left\{ x \in \mathbb{R}^n \mid \exists (\lambda_i)_{i=1,\ldots,N_v} \in [0,1]^{N_v} \; x = \sum_{k=1}^{N_v} \lambda(k) v(k) \text{ and } \sum_{k=1}^{N_v} \lambda(k) = 1 \right\} \tag{A.12}$$

This representation is called Vertex Representation or V-Representation.

Example:Polytope H-Rep and V-Rep

Figure.A.2 depicts the polytope $\mathcal{P}$ defined by its H-representation with the

intersection of two halfspaces:

$$\begin{cases} \mathcal{H}_1 : & -x_1 + x_2 \leq 4 \\ \mathcal{H}_2 : & x_2 \leq 3 \\ \mathcal{H}_3 : & x_1 \leq 4 \\ \mathcal{H}_4 : & \frac{1}{2}x_1 + x_2 \leq -2 \end{cases} \tag{A.13}$$

The polytope can also be define by its vertices given in the collection $\mathcal{V}$:

$$\mathcal{V} = \left\{ \begin{bmatrix} -1 \\ 3 \end{bmatrix}, \begin{bmatrix} 4 \\ 3 \end{bmatrix}, \begin{bmatrix} 4 \\ -4 \end{bmatrix}, \begin{bmatrix} -4 \\ 0 \end{bmatrix} \right\} \tag{A.14}$$

corresponding to the location of hyperplanes intersections as showed on Figure.A.3.

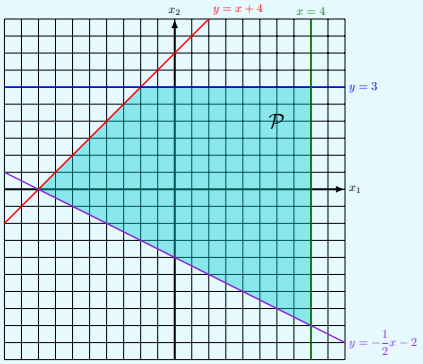

Figure A.2: Example of a Polytope defined by its H-Representation

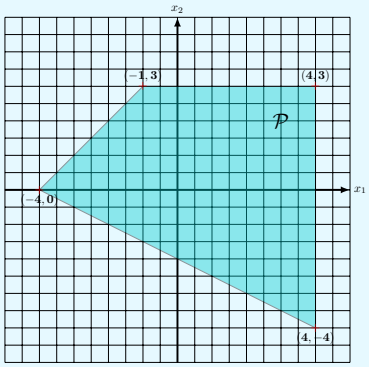

Figure A.3: Example of a Polytope defined by its V-Representation

Operation on convex sets

Proposition 1.1: Cartesian product of polyhedral sets

Given two polyhedrons $\mathcal{P}_1$ and $\mathcal{P}_2$ and their H-Representation:

$$\begin{aligned} \mathcal{P}_1 &= \{x \in \mathbb{R}^n \mid F_1 x \leq g_1\}, \; F_1 \in \mathbb{R}^{l \times n}, \; g_1 \in \mathbb{R}^l \\ \mathcal{P}_2 &= \{x \in \mathbb{R}^m \mid F_2 x \leq g_2\}, \; F_2 \in \mathbb{R}^{s \times m}, \; g_2 \in \mathbb{R}^s \end{aligned} \tag{A.15}$$

Then their cartesian product can be expressed as:

$$\mathcal{P}_1 \times \mathcal{P}_2 = \left\{ x \in \mathbb{R}^{n+m} \mid \begin{bmatrix} F_1 & 0 \\ 0 & F_2 \end{bmatrix} x \leq \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} \right\} \tag{A.16}$$

The cartesian product of polyhedrons allows,in the control framework, is computationally useful when considering two distincts dynamics (such as rails and hexapod actuations).

Definition 1.1: Minkowski sum

Let $X \subset \mathbb{R}^n$ and $Y \subset \mathbb{R}^n$. The Minkowski sum of X and Y, denoted by $\oplus$, is the set:

$$X \oplus Y = \{x + y \mid x \in X, y \in Y\} \tag{A.17}$$

The Minkowski sum operator allows the formalization of a displacement (or translation) of a polytope.

Proposition 1.2: Translation of polytopes

If $\mathcal{P} \subset \mathbb{R}^n$ is a convex polytope defined by its vertices $(v_1, \ldots, v_{N_v})$, then the translation of $\mathcal{P}$ along the vector $x \in \mathbb{R}^n$ can be expressed as

$$\begin{aligned} t(\mathcal{P}, x) &= \{x\} \oplus \mathcal{P} \\ &= Conv\{v_1 + x, \ldots, v_{N_v} + x\} \end{aligned} \tag{A.18}$$

Proposition 1.3: Scaling of a polytope

Let $\mathcal{P} \subset \mathbb{R}^n$ a convex polytope defined by its H-Representation:

$$\mathcal{P} = \{x \in \mathbb{R}^n \mid Fx \leq g\} \tag{A.19}$$

and $\lambda \in \mathbb{R}$. Then:

$$\lambda\mathcal{P} = \{x \in \mathbb{R}^n \mid Fx \leq \lambda g\} \tag{A.20}$$

If $\mathcal{P} \subset \mathbb{R}^n$ is a convex polytope defined by its vertices $(v_1, \ldots, v_{N_v})$, then $\lambda\mathcal{P}$ is defined by $(\lambda v_1, \ldots, \lambda v_{N_v})$.

Example:Translation and scaling

Considering the previous polytope $\mathcal{P}$, the Figure.A.4 depicts its tanslation according to the vector $x = [-3 \ -2]^T$ and its reduction through a scaling factor $\alpha = 0.2$.

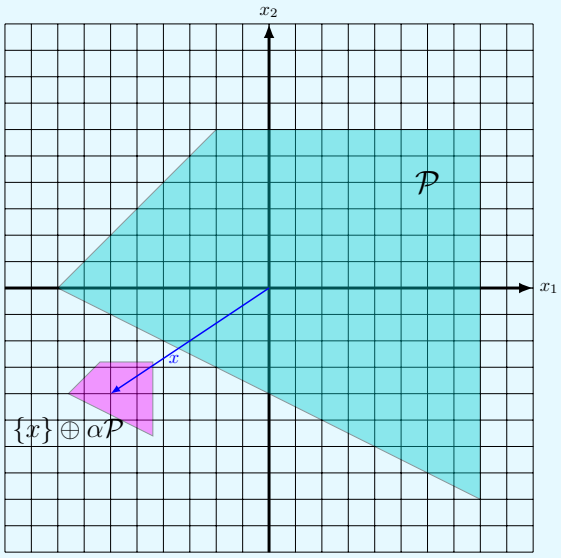

Figure A.4: Translation and scaling of a polytope

Definition A.13: Pontryagin difference

Let $X \subset \mathbb{R}^n$ and $Y \subset \mathbb{R}^n$. The Pontryagin difference of X and Y, denoted by $\ominus$, is the set:

$$X \ominus Y = \{x \in \mathbb{R}^n \mid \{x\} \oplus Y \subset X\} \tag{A.21}$$

Optimization

We introduce here the basics of convex optimization and the derived properties.

Definition A.14: Convex Optimization problem

Let $C \subset \mathbb{R}^n$. An **optimization problem** consists in minimizing a cost function $J : \mathbb{R}^n \mapsto \mathbb{R}$ under constraints on the optimization variables. It is denoted as follows:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & J(x) \\ \text{subject to} \quad & x \in C \end{aligned} \tag{A.22}$$

More, the optimization problem (A.22) is said to be **convex** if:

1. J is a convex function
2. C is a convex set

Remark. *The optimization may consists in maximizing a function which is equivalent to minimizing the opposite of the function.*

Proposition 1.4: local and global minimum of a convex optimization

If (A.22) is a convex optimization problem, then any local minimum is a global minimum.

Definition A.15: Feasibility

The optimization problem (A.22) is said to be **feasible** if the constraint set is nonempty ($C \neq \varnothing$)

Consequently, the convex optimization problem whose constraints set is nonempty is feasible and has a unique solution.

The taxonomy of optimization problem is rich but some of them are well known and their numerical resolution has been enhanced with mathematics, algorithmics and computationnal progress making them available for real-time implementation.

Definition A.16: Linear Programming (LP)

The optimization problem (A.22) is said to be linear if:

1. The cost function is linear, *i.e* $J(x) = a^T x,\ a \in \mathbb{R}^n$
2. The constraint set is a convex polytope (defines by its H-Representation)

Notation:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & a^T x \\ \text{subject to} \quad & Fx \leq g \end{aligned} \tag{A.23}$$

where $F \in \mathbb{R}^{m \times n}$ and $g \in \mathbb{R}^m$

Definition A.17: Quadratic Programming (QP)

The optimization problem (A.22) is said to be quadratic if:

1. The cost function is quadratic, *i.e* $J(x) = x^T Q x + a^T x$ with $Q \in \mathbb{R}^{n \times n},\ Q \succ 0,\ a \in \mathbb{R}^n$
2. The constraint set is a convex polytope (defines by its H-Representation)

Notation:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & x^T Q x + a^T x \\ \text{subject to} \quad & Fx \leq g \end{aligned} \tag{A.24}$$

where $F \in \mathbb{R}^{m \times n}$ and $g \in \mathbb{R}^m$

A.1.3 Tools for Optimal Control

In this thesis we mainly use states representation of dynamical systems and particularly the discrete-time framework of those ones because of the real-time requirements.

Definition A.18: State-space representation of discrete-time systems

Let $f : \mathbb{R}^n \mapsto \mathbb{R}^n$ and $g : \mathbb{R}^n \mapsto \mathbb{R}^m$. A discrete-time state space representation of a autonomous system is given by:

$$x(k+1) = f(x(k)) \tag{A.25}$$

The representation of a system with input is given by:

$$\begin{array}{rcl} x(k+1) & = & f(x(k), u(k)) \\ y(k) & = & g(x(k), u(k)) \end{array} \tag{A.26}$$

The representation of an uncertain system is given by:

$$\begin{array}{rcl} x(k+1) & = & f(x(k), u(k), w(k)) \\ y(k) & = & g(x(k), u(k), w(k)) \end{array} \tag{A.27}$$

where $w(.) \in \mathbb{R}^n$ is a disturbance.

Definition A.19: Linear Time Invariant (LTI) systems

Considering the previous notations, a system is said to be **Linear Time Invariant (LTI)** if the functions f and g are linear.

A LTI autonomous system is represented by:

$$x(k+1) = Ax(k),\ A \in \mathbb{R}^{n \times n} \tag{A.28}$$

A LTI system with input is represented by:

$$\begin{array}{rcl} x(k+1) & = & Ax(k) + Bu(k) \\ y(k) & = & Cx(k) + Du(k) \end{array} \tag{A.29}$$

Where $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, $C \in \mathbb{R}^{l \times n}$ and $D \in \mathbb{R}^{l \times m}$.

A LTI uncertain system is represented by:

$$\begin{array}{rcl} x(k+1) & = & Ax(k) + Bu(k) + w(k) \\ y(k) & = & Cx(k) + Du(k) + w(k) \end{array} \tag{A.30}$$

where $w(.) \in \mathbb{R}^n$ is a disturbance, $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, $C \in \mathbb{R}^{l \times n}$ and $D \in \mathbb{R}^{l \times m}$.

Definition A.20: Controllability

A LTI system represented by (A.29) is said to be **controllable** if the matrix:

$$\mathcal{C} = \begin{bmatrix} B & AB & A^2B & \dots & A^{n-1}B \end{bmatrix} \tag{A.31}$$

is full rank.

Definition A.21: Observability

A LTI system represented by (A.29) is said to be **observable** if the matrix:

$$\mathcal{O} = \begin{bmatrix} C & CA & CA^2 & \dots & CA^{n-1} \end{bmatrix}^T \tag{A.32}$$

is full rank.

Definition A.22: LQ control

We consider a LTI controllable system represented by (A.29). **LQ Control** consists in solving the following optimization problem:

$$\begin{array}{ll} \underset{(u_0, u_1, \dots)}{\text{minimize}} & \displaystyle\sum_{k=0}^{\infty} x(k)^T Q x(k) + u(k)^T R u(k) \\ \text{subject to} & x(k+1) = Ax(k) + Bu(k) \end{array} \tag{A.33}$$

where $Q \in \mathbb{R}^{n \times n}$ and $R \in \mathbb{R}^{m \times m}$ are symmetric.

On can demonstrate that the solution is given by the state feedback:

$$u(k) = -Kx(k) \tag{A.34}$$

where the linear gain K is given by:

$$K = (R + B^T P B)^{-1} B^T P A \tag{A.35}$$

and P is the solution of the discrete Riccati equation:

$$P = Q + A^T P A - (A^T P B)(R + B^T P B)^{-1}(B^T P A) \tag{A.36}$$

Example: LQ control and Maximal Positively Invariant Set

Consider the LTI system with the following state space representation:

$$x(k+1) = \begin{bmatrix} 1 & 0.5 \\ 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} 0.125 \\ 0.5 \end{bmatrix} u(k) \tag{A.37}$$

subject to the following constraints:

$$\begin{aligned} |x_1(k)| &\leq 2 \\ |x_2(k)| &\leq 4 \\ |u(k)| &\leq 5 \end{aligned} \tag{A.38}$$

Thanks to the LQ procedure given in Definition.A.22,we can compute the linear feedback law K with $Q = \begin{bmatrix} 500 & 0 \\ 0 & 100 \end{bmatrix}$ and $R = 50$.

$$K = \begin{bmatrix} 1.5639 & 1.9018 \end{bmatrix} \tag{A.39}$$

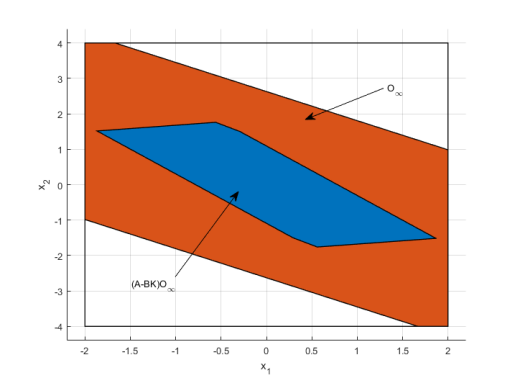

Figure A.5: Example of Maximal Positively Invariant Set O_∞ in red and its image through the feedback $A - BK$ (in blue)

Appendix B

Résumé en français

Contents

B.1 Introduction

B.1.1 Qu'est-ce que la simulation de conduite ?

Aujourd'hui, l'industrie automobile se développe de manière à améliorer:

- le confort de conduite
- la sécurité (prévention des accidents)
- la pollution (électrification, particules fines,...)
- le rapport qualité/prix des véhicules

Ainsi les constructeurs développent des nouveaux véhicules ou des systèmes d' assistance à la conduite (ADAS) et doivent garantir la satisfaction à des normes (de sécurité et d'écologie par exemple) et aux attentes des clients. Cependant, le nombre, la complexité et la dangerosité des différentes situations de conduite potentielles sont un sérieux obstacle à la validation des nouveaux prototypes. La simulation permet la reproduction artificielle de scénarios de conduite dans un environnement paramétrable, restreint et sécurisé. Le conducteur est et restera le centre de l'attention jusqu'à l'automatisation complète de la conduite. Ainsi, la perception des stimuli externes et le processus de prise de décision sont au coeur de la conduite. Une **experience de simulation** doit, par conséquent, reproduire les sensations du conducteur. Nous proposons la définition suivante pour la simulation de conduite

Definition B.1: Simulation de conduite

La simulation de conduite est l'ensemble des processus physiques et numériques permettant la reproduction des sensations du conducteur dans un environnement contraint et sécurisé.

Les sens connus à ce jour sont: la vue, l'ouïe, l'odorat, le goût, le toucher, la thermoception, la nociception, la proprioception et **l'équilibrioception**. Il est clair que certains de es sens sont plus important pour la conduite, le principal était la vue. Le conducteur ressent également les accélération via l'équilibrioception via le système vestibulaire dans l'oreille interne.

La restitution des accélérations est le principal sujet de cette thèse. Puisque l'accélération décrit des déplacements, un espace de travail d'une certaine taille est nécessaire. Si on considère un scénario de passage de 0 à 100km/h en $\Delta t = 10$s avec une accélération constante a_{veh}. La distance parcourue par le véhicule est estimée par intégration:

$$L = \frac{1}{2} a_{veh} \Delta t^2 \approx 140m \tag{B.1}$$

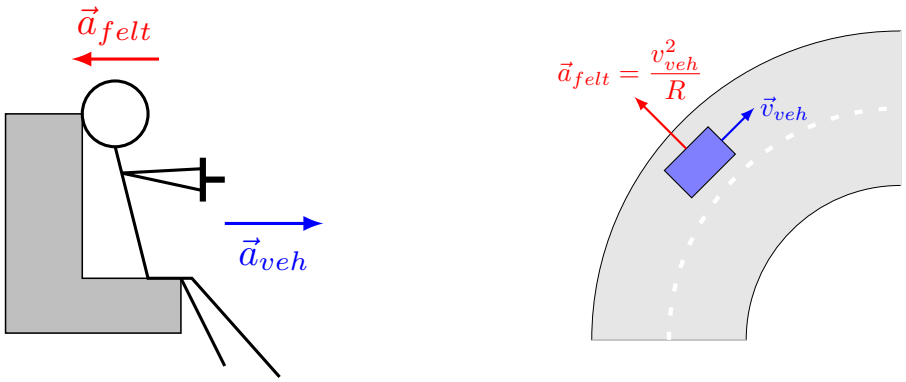

Figure B.1: Accélération ressentie longitudinalement (gauche) and latéralement dans un virage (droite)

Un tel scénario nécessite donc un espace de travail de très grande taille. Une solution pour économiser des déplacement est d'exploiter le champ gravitationnel en inclinant le conducteur et l'environnement virtuel de manière à projeter l'accélération gravitationnel sur le système vestibulaire du conducteur. Cette technique est appelée **"Tilt Coordination"** dans la littérature et illustrée en Figure B.2

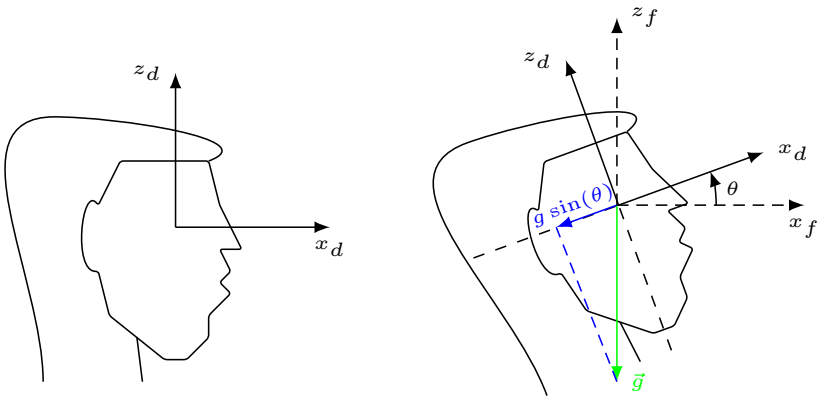

Figure B.2: Illustration de la technique de "Tilt Coordination"

B.1.2 Les simulateurs de conduite

Les simulateurs peuvent être statiques ou dynamique, dans cette thèse nous nous intéressons spécifiquement aux **simulateurs dynamiques** équipés de système de mise en mouvement. La composition de tels simulateurs est montrée Figure B.3 à travers l'exemple du simulateur de Renault ULTIMATE.

Ce genre de simulateur est constitué:

- d'un écran panoramique où l'environnement virtuel est affiché
- une cabine de voiture équipée de sièges, de pédales, d'un tableau de bord, etc...
- un hexapode (ou plateforme de Stewart) qui incline la cabine pour la technique de "Tilt Coordination"
- Des rails mettant en mouvement la plateforme latéralement et longitudinalement pour générer des mouvements rapides.

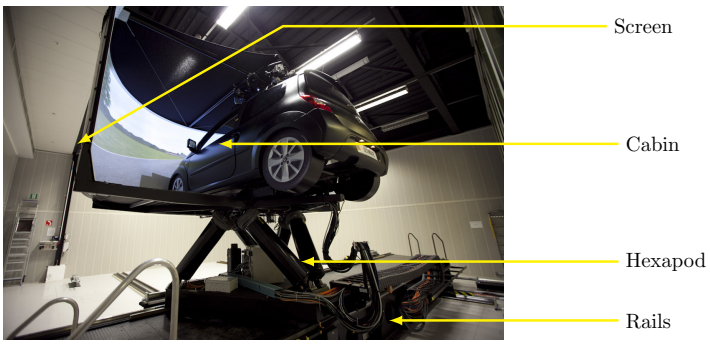

Figure B.3: Composition d'un simulateur dynamique (Renault ULTIMATE)

Le fonctionnement de ce type de simulateur est résumé en Figure B.4

1. Le conducteur dans la cabine réagit aux stimulis visuels affichés à l'écran en appuyant sur la pédale d'accélération ou de frein et en tournant le volant.
2. Les signaux générés sont traités par le logiciel de simulation. Ce dernier calcule ainsi les accélérations que devraient ressentir le conducteur.
3. Les signaux d'accélération sont des références à suivre pour l'architecture de contrôle de la plateforme en temps réel. Les actionneurs (rails et hexapode) se déplacent ainsi de manière à restituer au mieux les sensations attendues par le conducteur.

Le sujet de la thèse est la structure de commande des simulateurs dynamiques et en particulier la fonction traduisant les accélérations à ressentir en profil de déplacement de la plateforme de simulation appelée **"Motion Cueing Algorithm" (MCA)**.

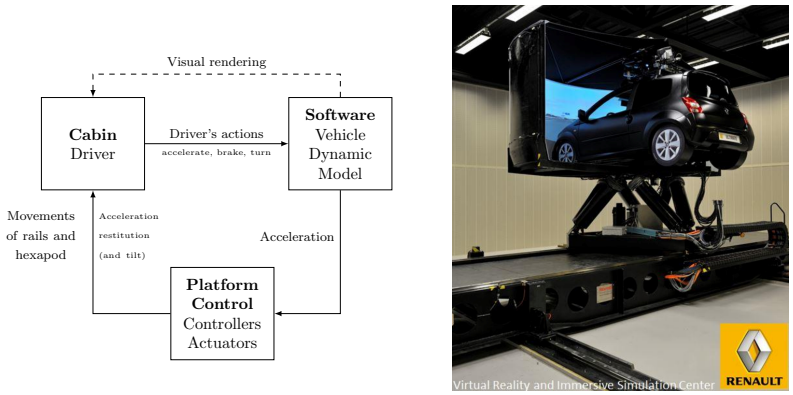

Figure B.4: Gauche: Schéma de fonctionnement d'un simulateur dynamique. Droite: Simulateur Renault ULTIMATE

B.2 Algorithme de restitution des accélérations: MCA

B.2.1 L'architecture de commande et l'accélération ressentie

Architecture de commande

L'architecture de commande du simulateur est résumée Figure B.5. Les signaux d'accélération à restituer au conducteur calculés par le logiciel sont traités par le MCA (Motion Cueing Algorithm) qui calcule un profil de position pour les rails et d'inclinaison pour l'hexapode. Les actionneurs se déplacent ainsi de manière à restituer au mieux les accélérations.

Accélération ressentie

L'accélération ressentie par le conducteur dans le simulateur est la contribution de l'accélération d'entraînement des rails et de la projection du champ gravitationnel sur le système vestibulaire du conducteur. On peut modéliser cette accélération ressentie par l'équation (B.2) grâce au principe fondamental de la dynamique dans le référentiel du conducteur (Figure B.6).

$$a_{felt} = a_{lin}\cos(\theta) + g\sin(\theta) \tag{B.2}$$

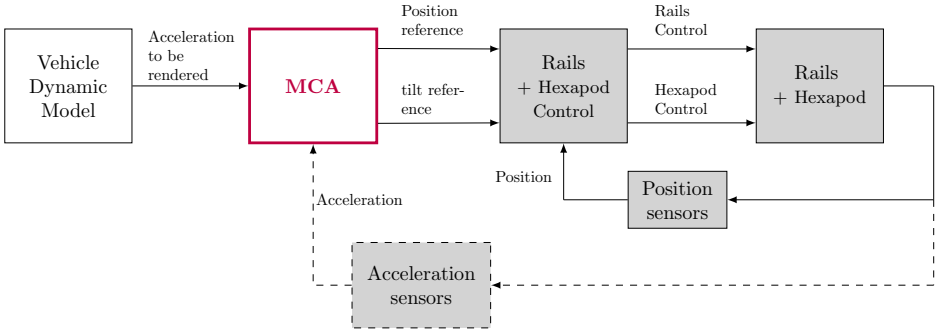

Figure B.5: Schéma bloc de la structure de contrôle d'un simulateur dynamique. Les éléments grisés sont ceux des composants des fournisseurs qui ne peuvent pas être modifié à notre niveau

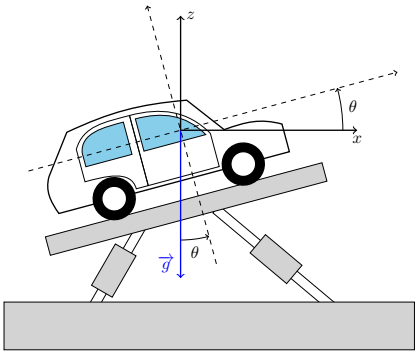

Figure B.6: Schéma vue de côté d'un simulateur dynamique

B.2.2 Structure par filtres

Le type d'architecture la plus commune dans l'industrie et la plus simple à mettre en oeuvre est la structure par filtre (Figure B.7). La philosophie de cette technique est de séparer les mouvements rapides (hautes fréquences) des mouvements lents (basses fréquences), les premiers sont directement restitués par les rails alors que les secondes sont restitués par l'inclinaison de l'hexapode. Cette technique présente de faibles performances mais cet inconvénient doit être mis en balance avec sa simplicité d'implémentation qui ne nécessite pas de connaissance particulière dans le domaine de l'automatique car c'est un design en boucle ouverte.

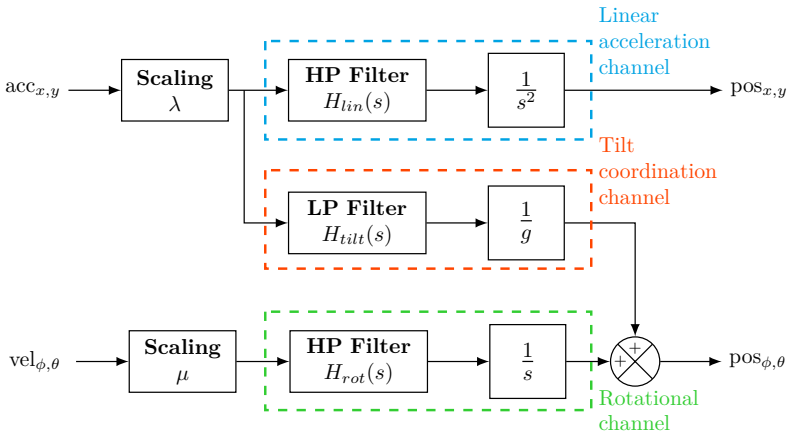

Figure B.7: Structure de MCA à base de filtrage

B.2.3 MCA par commande prédictive

Modélisation

L'utilisation de la commande optimal implique l'utilisation de représentation d'état. Comme MCA est une fonction transformant les accélérations en positions sous contraintes, il peut être modélisé par un double intégrateur (pour les rails et pour l'hexapode). Ainsi, en choisissant le vecteur d'état: $x(k) = \begin{bmatrix} p(k) & v(k) & \theta(k) & \Omega(k) \end{bmatrix}^T$ et le vecteur d'entrée: $u(k) = \begin{bmatrix} a(k) & \gamma(k) \end{bmatrix}^T$:

$$x(k+1) = \begin{bmatrix} 1 & T_s & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T_s \\ 0 & 0 & 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} \frac{T_s^2}{2} & 0 \\ T_s & 0 \\ 0 & \frac{T_s^2}{2} \\ 0 & T_s \end{bmatrix} u(k) \tag{B.3}$$

$$y(k) = \begin{bmatrix} 0 & 0 & g & 0 \end{bmatrix} x(k) + \begin{bmatrix} 1 & 0 \end{bmatrix} u(k)$$

où $p(k)$, $v(k)$ et $a(k)$ sont respectivement la position, la vitesse et l'accélération de la plateforme par rapport aux rails alors que $\theta(k)$, $\Omega(k)$ et $\gamma(k)$ sont respectivement l'angle d'inclinaison, la vitesse et l'accélération angulaire. La sortie $y(k)$ est l'accélération ressentie.

Cette modélisation est un exemple parmi d'autre, dans la thèse nous utilisons

la formalisation classique:

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= Cx(k) + Du(k) \end{aligned} \tag{B.4}$$

Le contrôleur MPC résout le problème d'optimisation suivant en boucle fermé:

$$\begin{aligned} \underset{(u(k),\ldots,u(k+N-1))}{\text{minimize}} \quad & \sum_{i=1}^{N} \|y^{ref}(k+i) - y(k+i)\|_{Q_y}^2 + \|x(k+i)\|_{Q_x}^2 \\ & + \|u(k+i)\|_R^2 + \|x(k+N)\|_P^2 \\ \text{subject to} & \\ x(k+i+1) = Ax(k+i) + Bu(k+i) \quad & \forall i \in \{0,\ldots,N-1\}, \\ y(k+i) = Cx(k+i) + Du(k+i) \quad & \forall i \in \{0,\ldots,N\}, \\ x(k+i) \in \mathcal{X} \quad & \forall i \in \{1,\ldots,N\}, \\ u(k+i) \in \mathcal{U} \quad & \forall i \in \{0,\ldots,N-1\}, \\ x(k+N) \in \mathcal{X}_f & \end{aligned} \tag{B.5}$$

où $\mathcal{X}_f$ est un ensemble invariant positif. Le principe de la commande prédictive est rappelé Figure B.8. L'avantage de cette technique est qu'elle est potentiellement

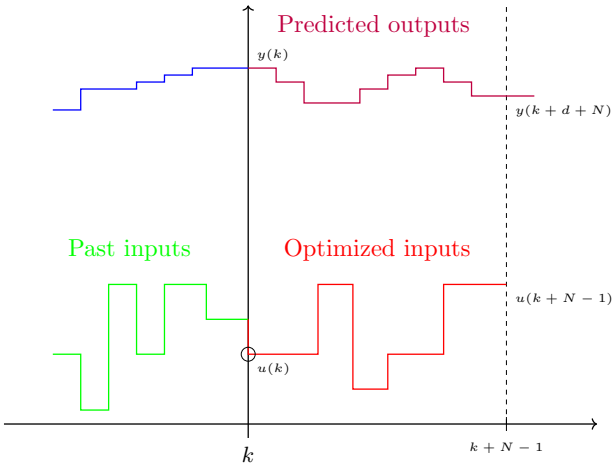

Figure B.8: Fonctionnement de la commande prédictive: le contrôleur trouve la séquence optimale d'entrées (en rouge) qui génère une trajectoire prédite de la sortie (en violet), enfin seule la première composante de la séquence d'entrée est appliquée au système.

performante en fonction de la qualité de la prédiction et de la longueur de l'horizon

de prédiction. Cependant, l'implémentation temps réel peut-être compliquée à réaliser pour des longs horizons. De même la prédiction est très compliquée à obtenir. La prédiction et l'implémentation temps réel sont des problèmes ouverts de ce domaine de recherche.

B.3 MPC améliorées pour MCA

B.3.1 MPC non linéaire

La première contribution de cette thèse est la prise en compte des non-linéarités dues à la projection du champ gravitationnel (équation (B.6)).

$$x(k+1) = \begin{bmatrix} p(k) + T_s v(k) + \frac{T_s^2}{2} u_l(k) \\ v(k) + T_s u_l(k) \\ \theta(k) + T_s u_r(k) \end{bmatrix} = f(x(k), u(k)) \tag{B.6}$$

$$y(k) = g \sin(\theta(k)) + u_l(k) \cos(\theta(k)) = g(x(k), u(k))$$

Ainsi, le problème d'optimisation associé au contrôleur MPC devient non linéaire et donc moins adapté à l'implémentation temps réel:

$$\begin{aligned} \underset{(u(k),\ldots,u(k+N-1))}{\text{minimize}} \quad & \sum_{i=1}^{N} \|y^{ref}(k+i) - y(k+i)\|_{Q_y}^2 + \|x(k+i)\|_{Q_x}^2 \\ & + \|u(k+i)\|_R^2 + \|x(k+N)\|_P^2 \\ \text{subject to} & \\ x(k+i+1) = f(x(k+i), u(k+i)) \quad & \forall i \in \{0, \ldots, N-1\}, \\ y(k+i) = g(x(k+i), u(k+i)) \quad & \forall i \in \{0, \ldots, N\}, \\ x(k+i) \in \mathcal{X} \quad & \forall i \in \{1, \ldots, N\}, \\ u(k+i) \in \mathcal{U} \quad & \forall i \in \{0, \ldots, N-1\}, \\ x(k+N) \in \mathcal{X}_f & \end{aligned} \tag{B.7}$$

Nous proposons dans cette contribution de trouver une alternative compatible avec le temps réel pour ce type de contrôleur prédictif en comparant deux contrôleurs:

1. L-NL (Linear-NonLinear): Compensation de la réponse des rails par celle de l'inclinaison (Figure B.9).

2. NL-L (NonLinear-Linear): Compensation de la réponse de l'hexapode par celle des rails (Figure B.10).

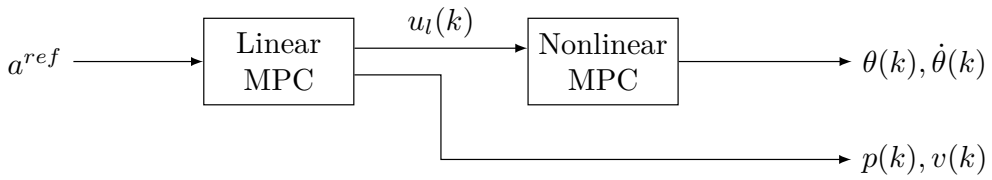

Figure B.9: principe de la stratégie L-NL

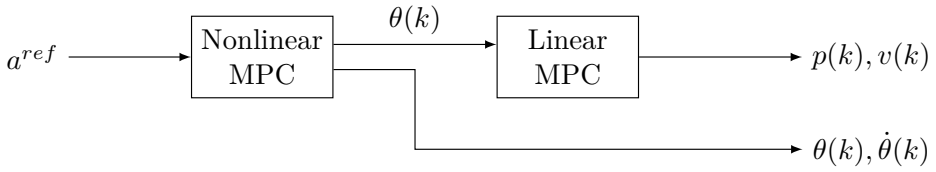

Figure B.10: principe de la stratégie NL-L

La comparaison des performances et du temps de calcul montre que le contrôleur L-NL est le plus adapté pour une implémentation temps-réel.

B.3.2 Compensation du retard

La deuxième contribution concerne la compensation des retards dus à l'inertie de la plateforme, au protocole de communication ou aux erreurs de prediction. On considère dans cette partie le modèle linéaire à retard suivant:

$$x(k+1) = \begin{bmatrix} 1 & T_s & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} \frac{T_s^2}{2} & 0 \\ T_s & 0 \\ 0 & T_s \end{bmatrix} u(k-d)$$
$$y(k) = \begin{bmatrix} 0 & 0 & g \end{bmatrix} x(k) + \begin{bmatrix} 1 & 0 \end{bmatrix} u(k-d) \tag{B.8}$$

Une manière d'aborder le contrôle de tels systèmes est de considérer un espace d'état étendu comprenant l'historique des commandes passées. Ainsi la dynamique étendue peut être formalisée par un modèle linéaire non retardé, par conséquent un contrôleur MPC linéaire peut être considéré. Cependant, la grande dimension de l'espace d'état et la difficulté de calculer un ensemble invariant terminal fait que cette technique n'est pas adaptée pour le temps réel dans notre application. La philosophie de cette contribution est d'utiliser une formalisation de MPC allégée

utilisant l'ensemble maximal contrôlable (B.9). Cette stratégie est résumée Figure B.11.

$$
\begin{aligned}
\underset{(u(k),\ldots,u(k+N-1))}{\text{minimize}} \quad & \sum_{i=d+1}^{d+N} \|y^{ref}(k+i) - y(k+i)\|_{q_y}^2 + \|x(k+i)\|_{Q_x}^2 + \sum_{i=1}^{N} \|u(k+i)\|_R^2 \\
\text{subject to} \quad & x(k+i+1) = Ax(k+i) + Bu(k+i-d), \\
& \qquad\qquad \forall i \in \{0,\ldots,d+N-1\}, \\
& y(k+i) = Cx(k+i) + Du(k+i-d), \\
& \qquad\qquad \forall i \in \{d+1,\ldots,d+N\}, \\
& x(k+d+1) \in \mathcal{C}, \\
& (u(k),\ldots,u(k+N-1)) \in \mathcal{U}
\end{aligned}
\tag{B.9}
$$

where q_y, Q_x and R are weighting matrices.

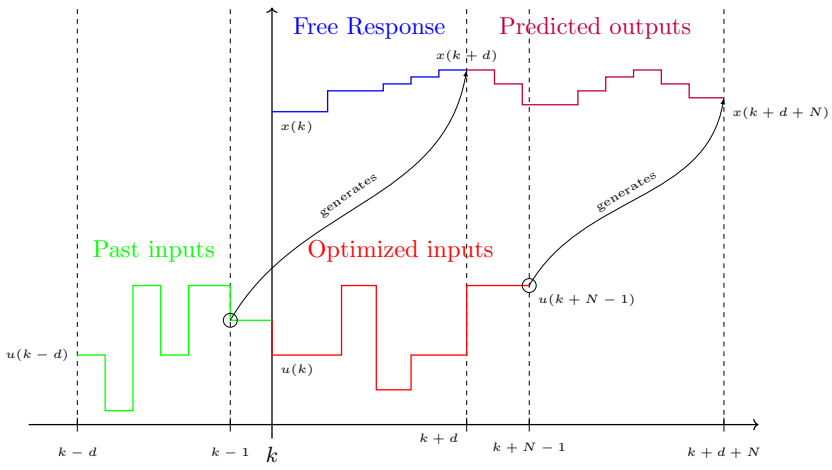

Figure B.11: Principe de la stratégie: Les commandes passées imposent une réponse inertielle sur $k+d$ étapes alors que les commandes optimisées sont choisis tel que les entrées prédites assurent l'appartenance du premier état prédit (à l'étape $k+d+1$) à l'ensemble maximal contrôlable $\mathcal{C}$.

B.4 Commmande par interpolation pour MCA

B.4.1 Commande par interpolation: le cas de la régulation (IBC)

On considère le système linéaire discret:

$$x(k+1) = Ax(k) + Bu(k) \tag{B.10}$$

avec $x \in \mathcal{X} \subset \mathbb{R}^n, u \in \mathcal{U} \subset \mathbb{R}^m$, $A \in \mathbb{R}^{n\times n}, B \in \mathbb{R}^{n\times m}$. $\mathcal{X}$ et $\mathcal{U}$ sont des ensembles de contraintes sur les états et sur les entrées contenant l'origine.

On considère deux ensembles contrôlables $\Omega^o \subset \mathcal{X}$ et $\Omega^v \subset \mathcal{X}$ convexes compacts contenant l'origine tels que:

$$\Omega^o \subset \Omega^v \subset \mathcal{X} \tag{B.11}$$

Ω^o est appelé *ensemble interne* et Ω^v l'*ensemble externe*. Enfin, chacun est associé à sa propre loi de commande contrainte:

$$u^o = \mathcal{K}^o(x) \text{ si } x \in \Omega^o \text{ et } u^v = \mathcal{K}^v(x) \text{ si } x \in \Omega^v/\Omega^o \tag{B.12}$$

tel que $\mathcal{K}^o(\Omega^o) \subset \mathcal{U}$ et $\mathcal{K}^v(\Omega^v) \subset \mathcal{U}$.

Le principe de la commande par interpolation est le suivant:

1. l'état courant $x(k)$ peut être décomposé de manière convexe:

$$x(k) = c(k)x^v(k) + (1 - c(k))x^o(k) \tag{B.13}$$

où $x^o(k) \in \Omega^o$, $x^v(k) \in \Omega^v$ et $c(k)$ est un facteur de convexité choisi pour maximiser la contribution du contrôleur local de Ω^o via la résolution du problème d'optimisation suivant:

$$\begin{aligned} \begin{bmatrix} x^v & x^o & c^* \end{bmatrix}^T = \underset{(x^v,x^o,c)(k)}{\text{minimize}} \quad & c \\ \text{subject to} \quad & x^v \in \Omega^v, x^o \in \Omega^o, \\ & x(k) = cx^v + (1-c)x^o, \\ & c \in [0,1]. \end{aligned} \tag{B.14}$$

Une interprétation géométrique de cette décomposition est donnée Figure B.12

2. La commande appliquée au système est la combinaison convexe des deux retours d'états comme explicité sur le schéma block Figure B.13

$$u(k) = c(k)u^v(k) + (1 - c(k))u^o(k) \tag{B.15}$$

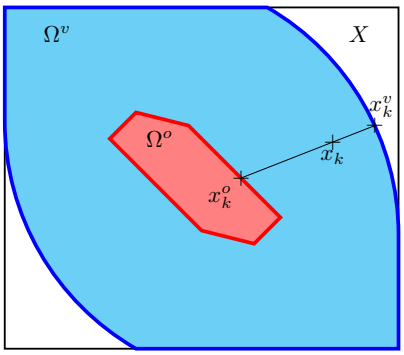

Figure B.12: Interprétation géométrique de la décomposition convexe de l'état courant $x(k)$

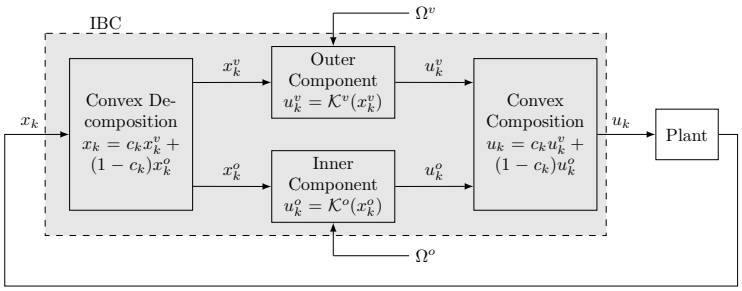

Figure B.13: Principe de la commande par interpolation

Cette stratégie de commande est prouvée stable et récursivement faisable, de plus son potentiel computationnel plus intéressant qu'une stratégie MPC à long horizon en fait une bonne stratégie candidate pour notre application. Nous devons cependant l'adapter dans un premier temps à la poursuite de trajectoires.

B.4.2 Poursuite de trajectoire dynamique par interpolation (IBT)

Pour adapter la commande par interpolation à la poursuite de trajectoire, nous proposons d'utiliser un guide de référence (Reference Governor) calculant une référence admissible qui pourra être suivi grace à la commande par interpolation comme résumé par le schéma block Figure B.14

Le guide de référence calcule une nouvelle référence faisable et une topologie contrôlable dans laquelle la philosophie de la commande par interpolation est possible. Cette topologie est déterminée par translation et mise à l'échelle de la

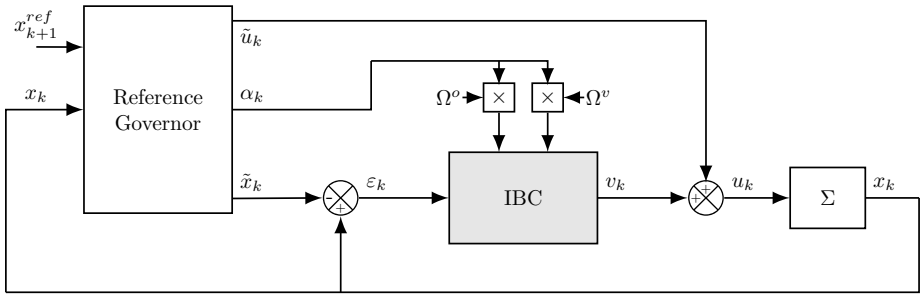

Figure B.14: Schéma bloc de la poursuite de trajectoire basée sur la commande par interpolation

topologie initiale via la résolution du problème d'optimisation (B.16).

$$\begin{aligned}\begin{bmatrix}\tilde{u}(k)\\ \alpha(k)\end{bmatrix} = \underset{(\tilde{u}(k),\alpha(k))}{\arg\min} \quad & \|x^{ref}(k+1) - \tilde{x}(k+1|k)\|_Q^2 \\ \text{subject to} \quad & \tilde{x}(k+1|k) = A\tilde{x}(k|k) + B\tilde{u}(k), \\ & x(k) \in \{\tilde{x}(k|k)\} \oplus \alpha(k)\Omega^v, \\ & \{\tilde{x}(k|k)\} \oplus \alpha(k)\Omega^v \subset \Omega^v, \\ & \tilde{u}(k) \in (1-\alpha(k))\mathcal{U}, \\ & \{\tilde{x}(k+1|k)\} \oplus \alpha(k)\Omega^v \subset \Omega^v\end{aligned} \tag{B.16}$$

Ainsi, la procédure de la commande par interpolation est applicable à l'erreur de poursuite: $\varepsilon(k) = x(k) - \tilde{x}(k)$ dans $\alpha(k)\Omega^v$.

La commande appliquée au système est la contribution du guide de référence et de la commande par interpolation.

$$u(k) = \tilde{u}(k) + \underbrace{c(k)v^v(\varepsilon^v(k)) + (1-c(k))v^o(\varepsilon^o(k))}_{v(k)} \tag{B.17}$$

où $v^v(k)$ et $v^o(k)$ sont les actions de commandes dans $\alpha(k)\Omega^v$ et $\alpha(k)\Omega^o$.

Cette thèse démontre la faisabilité récursive de cette stratégie de poursuite, ainsi que ses possibilités d'application dans la simulation de conduite comme stratégie de MCA.

B.5 p-Invariance

La dernière contribution de cette thèse concerne la relaxation des concept d'invariance positives et contrôlées et le potentiel allègement computationnel pour des contrôleurs optimaux.

On autorise ainsi une trajectoire à quitter un ensemble pendant un temps fini. On peut distinguer deux catégories de propriétés que l'on désignera par les termes "faibles" et "fortes".

Les propriétés faibles autorise le retour de la trajectoire avant le délai limite p alors que les propriétés fortes imposent le retour de la trajectoire au moment du délai p.

B.5.1 Le cas des systèmes autonomes

Dans cette partie nous nous intéressons aux systèmes autonomes décrits par:

$$x(k+1) = f(x(k)) \tag{B.18}$$

La faible satisfaction des contraintes est définie ci-dessous:

Definition B.2: p-satisfaction faible d'une contrainte

Soit $p \in \mathbb{N}^*$, la trajectoire du système (B.18) initialement en $x_0 \in \mathbb{R}^n$ p-satisfait faiblement la constrainte $\mathcal{L}(h)$ si il existe une fonction $r : \mathbb{N} \to \mathbb{N}_{[1,p]}$ tel que $x(k + r(k)) \in \mathcal{L}(h)$ pour tout $k \in \mathbb{N}$.

Cette définition est illustrée Figure B.15

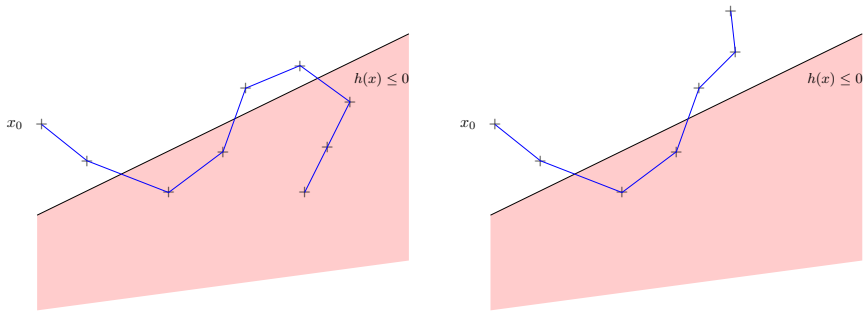

Figure B.15: Gauche: la trajectoire satisfait la contrainte $h(x) \leq 0$ avec un indice de 3. Droite: Pas de 3-satisfaction de la constrainte

Dans cette thèse nous utilisons cet outil d'analyse pour étendre cette propriété à des contraintes multiples et aux tubes de trajectoires. Nous pouvons ainsi définir la p-invariance faible pour un ensemble fini.

Definition B.3: Invariance faible

Soit $p \in \mathbb{N}$. Un ensemble $\Omega \subset \mathbb{R}^n$ est faiblement p-invariant pour le système (B.18) si pour tout $x_0 \in \Omega$, il existe une fonction $r : \mathbb{N} \to \mathbb{N}_{[1,p]}$ tel que $x(k + r(k)) \in \Omega$ pour tout $k \in \mathbb{N}$.

De la même façon, on définit les propriétés fortes de satisfaction et d'invariance des contraintes:

Definition B.4: p-satisfaction forte de contraintes multiples

Le tube de trajectoires du système (B.18) initialisé en $X \subset \mathbb{R}^n$ p-satisfait fortement le vecteur de contraintes $\mathcal{L}(h)$ avec $h : \mathbb{R}^n \to \mathbb{R}^m$ si $x(k + p) \in \mathcal{L}(h_i)$ $\forall i \in \mathbb{N}_{[1,m]}$, pour tout $x_0 \in X$ et pour tout $k \in \mathbb{N}$ tel que $x(k) \in X$ et $x(k + 1) \notin X$.

Cette définition est illustrée Figure B.16

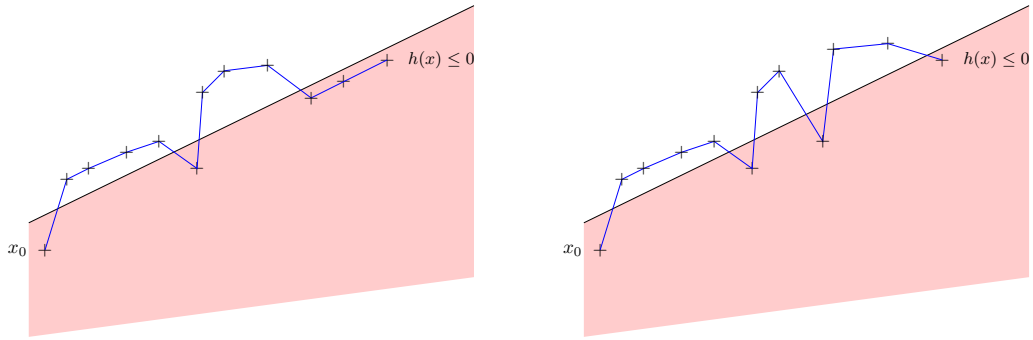

Figure B.16: Gauche: Illustration d'une trajectoire initiée en x_0 qui p-satisfait fortement la contrainte $h(x) \leq 0$ pour $p = 5$). Droite: exemple d'une trajectoire initiée en x_0 qui p-satisfait faiblement la contrainte $h(x) \leq 0$ mais ne satisfait pas fortement la même contrainte avec le même indice.

B.5.2 Cas des systèmes contrôlés

On considère dans cette partie les systèmes contrôlés contraints décrit par une dynamique linéaire:

$$x(k+1) = Ax(k) + Bu(k), \quad \text{tel que: } \begin{cases} x(k) \in \mathcal{X}, & \forall k, \\ u(k) \in \mathcal{U}, & \forall k \end{cases} \tag{B.19}$$

Les propriétés de p-invariance précédemment définies peuvent être étendues à ce type de système avec par exemple la p-invariance forte:

Definition B.5: p-invariance forte contrôlée

Un ensemble $\mathcal{B} \subset \mathcal{X}$ contenant l'origine est dit *fortement p-invariant* pour le système constrained system (B.19) si il existe $p \in \mathbb{N}^*$ tel que pour tout état $x(k) \in \mathcal{B}$, il existe une des options suivantes:

- une action de commande $u(k)$ tel que $x(k+1) \in \mathcal{B}$
- une séquence de commande $(u(k), ..., u(k+p-1)) \in \mathcal{U}^p$ tel que $x(k+p) \in \mathcal{B}$ et $(x(k+1), ..., x(k+p-1)) \in \mathcal{X}$.

Nous proposons dans cette thèse des stratégie de commande basée cette propriété dans le but d'utiliser la commande prédictive allégées manipulant des ensembles simples. Un prototype d'une telle technique est explicitée ci-dessous via le problème d'optimisation paramétré $\mathcal{O}(N_h, p, x(k))$:

$$\begin{aligned} J_p(x(k)) = \underset{(u(k),\ldots,u(k+N_h-1))}{\text{minimize}} \quad & \sum_{i=1}^{N_h} \|x^{ref}(k+i) - x(k+i)\|_Q^2 \\ \text{subject to} \quad & u(k+i) \in \mathcal{U} \ , x(k+i) \in \mathcal{X} \ \forall i \in [1, \ldots, M], , \\ & x(k+p) \in \mathcal{B}, \end{aligned} \tag{B.20}$$

with $M = \max(N_h, p)$. Bien que la résolution de ce problème d'optimisation n'est pas récursivement faisable, il est possible de l'utiliser dans l'Algorithme 4.

Dans cette thèse, nous prouvons que cet algorithme est récursivement faisable et peut être utilisé comme base pour une stratégie de MCA.

B.6 Conclusion

Cette thèse propose des alternatives aux stratégies de contrôle de plateformes de simulation de conduite existantes particulièrement celles basées sur l'optimisation comme la commande prédictive. L'utilisation de telles lois de commande en temps réel avec un faible temps d'échantillonnage rend leur implémentation difficile. Bien qu'il soit possible d'éviter cette problématique, cela se fait au prix d'une baisse des performances et d'une plus grande difficulté de paramétrage du contrôleur.

Les différentes contributions de cette thèse tentent de faire face à cette contrainte du temps réel tout en garantissant des performances de restitution sensorielle et une simplicité de paramétrage.

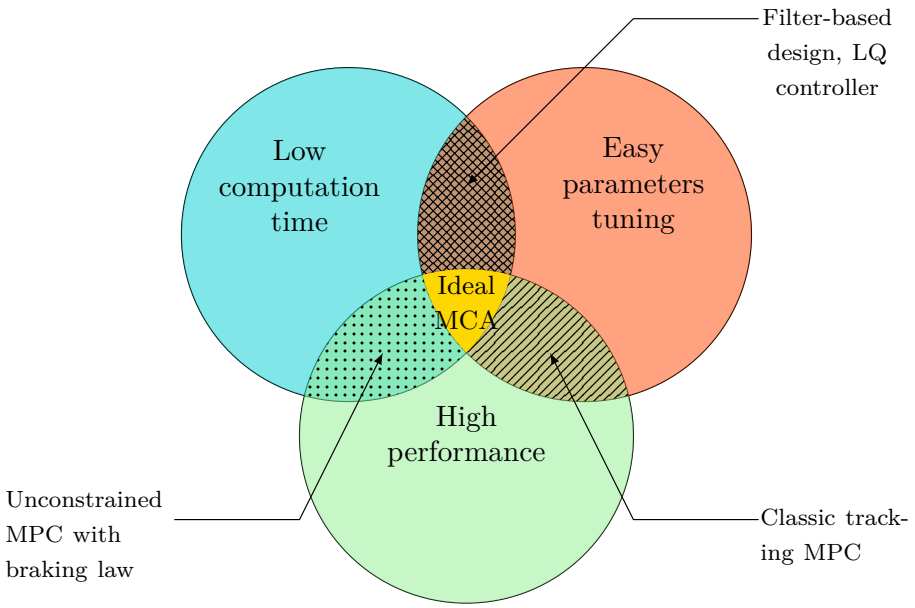

Figure B.17: Diagramme de Venn illustrant les enjeux structurels dans la conception du MCA

List of Figures

List of Tables

Bibliography

[Ariel and Sivan, 1984] Ariel, D. and Sivan, R. (1984). False cue reduction in moving flight simulators. *IEEE Transactions on Systems, Man, and Cybernetics*, (4):665–671.

[Aristotle, BCE] Aristotle (BCE). *De anima*.

[Asadi et al., 2016] Asadi, H., Mohamed, S., Lim, C. P., and Nahavandi, S. (2016). A review on otolith models in human perception. *Behavioural brain research*, 309:67–76.

[Athanasopoulos et al., 2014] Athanasopoulos, N., Bitsoris, G., and Lazar, M. (2014). Construction of invariant polytopic sets with specified complexity. *International Journal of Control*, 87(8):1681–1693.

[Athanasopoulos et al., 2013] Athanasopoulos, N., Doban, A. I., and Lazar, M. (2013). On constrained stabilization of discrete-time linear systems. In *21st Mediterranean Conference on Control and Automation*, pages 830–839. IEEE.

[Bacic et al., 2003] Bacic, M., Cannon, M., Lee, Y. I., and Kouvaritakis, B. (2003). General interpolation in mpc and its advantages. *IEEE Transactions on Automatic Control*, 48(6):1092–1096.

[Ballesteros-Tolosana et al., 2016] Ballesteros-Tolosana, I., Olaru, S., Rodriguez-Ayerbe, P., Pita-Gil, G., and Deborne, R. (2016). Comparison of optimization-based strategies for constrained control of auto-steering systems. In *2016 European Control Conference (ECC)*, pages 1592–1597. IEEE.

[Baotić, 2009] Baotić, M. (2009). Polytopic computations in constrained optimal control. *Automatika*, 50(3-4):119–134.

[Bemporad, 1998] Bemporad, A. (1998). Reference governor for constrained nonlinear systems. *IEEE Transactions on Automatic Control*, 43(3):415–419.

[Bitsoris, 1988] Bitsoris, G. (1988). Positively invariant polyhedral sets of discrete-time linear systems. *International Journal of Control*, 47(6):1713–1726.

[Blanchini and Miani, 2000] Blanchini, F. and Miani, S. (2000). Any domain of attraction for a linear constrained system is a tracking domain of attraction. *SIAM Journal on Control and Optimization*, 38(3):971–994.

[Blanchini and Miani, 2008] Blanchini, F. and Miani, S. (2008). *Set-theoretic methods in control*. Springer.

[Chisci and Zappa, 2003] Chisci, L. and Zappa, G. (2003). Dual mode predictive tracking of piecewise constant references for constrained linear systems. *International Journal of Control*, 76(1):61–72.

[Craig et al., 2009] Craig, A. B., Sherman, W. R., and Will, J. D. (2009). *Developing virtual reality applications: Foundations of effective design*. Morgan Kaufmann.

[Dagdelen et al., 2009] Dagdelen, M., Reymond, G., Kemeny, A., Bordier, M., and Maïzi, N. (2009). Model-based predictive motion cueing strategy for vehicle driving simulators. *Control Engineering Practice*, 17(9):995–1003. Publisher: Elsevier.

[De Angelo, 2000] De Angelo, J. (2000). The link flight trainer: A historic mechanical engineering landmark, roberson museum and science center, binghamton, new york, june 10, 2000.

[Dorea and Hennet, 1999] Dorea, C. E. T. and Hennet, J. (1999). (A, B)-invariant polyhedral sets of linear discrete-time systems. *Journal of Optimization Theory and Applications*, 103(3):521–542.

[Drosdol and Panik, 1985] Drosdol, J. and Panik, F. (1985). The daimler-benz driving simulator a tool for vehicle development. *SAE Transactions*, pages 981–997.

[Ellensohn et al., 2019] Ellensohn, F., Hristakiev, D., Schwienbacher, M., Venrooij, J., and Rixen, D. (2019). Evaluation of an optimization based motion cueing algorithm suitable for online application. In Kemeny, A., Colombet, F., Merienne, F., and Espié, S., editors, *Proceedings of the Driving Simulation Conference 2019 Europe VR*, pages 93–100, Strasbourg, France. Driving Simulation Association.

[Elloumi, 2006] Elloumi, H. (2006). *Commande des plates-formes avancées de simulation de conduite*. These de doctorat, Paris, ENMP.

[Falugi, 2015] Falugi, P. (2015). Model predictive control for tracking randomly varying references. *International Journal of Control*, 88(4):745–753.

[Fang and Kemeny, 2012a] Fang, Z. and Kemeny, A. (2012a). Motion cueing algorithms for a real-time automobile driving simulator. In *Driving Simulation Conference*, pages 159–174.

[Fang and Kemeny, 2012b] Fang, Z. and Kemeny, A. (2012b). Motion cueing algorithms for a real-time automobile driving simulator. In *Driving Simulation Conference*, pages 159–174.

[Fang and Kemeny, 2014] Fang, Z. and Kemeny, A. (2014). Review and prospects of Renault's MPC based motion cueing algorithm for driving simulator. In *Proceedings of the Driving Simulation Conference Europe, Paris, France*, pages 4–5.

[Fang and Kemeny, 2016] Fang, Z. and Kemeny, A. (2016). An efficient Model Predictive Control-based motion cueing algorithm for the driving simulator. *Simulation*, 92(11):1025–1033. Publisher: SAGE Publications Sage UK: London, England.

[Fang et al., 2010] Fang, Z., Reymond, G., and Kemeny, A. (2010). Performance identification and compensation of simulator motion cueing delays. *Trends in driving simulation design and experiments*, pages 111–120.

[Fang et al., 2019] Fang, Z., Wautier, D., and Kemeny, A. (2019). Development and applications of a fast mpc based motion cueing algorithm. In Kemeny, A., Colombet, F., Merienne, F., and Espié, S., editors, *Proceedings of the Driving Simulation Conference 2019 Europe VR*, pages 109–116, Strasbourg, France. Driving Simulation Association.

[Fernandez and Goldberg, 1971] Fernandez, C. and Goldberg, J. M. (1971). Physiology of peripheral neurons innervating semicircular canals of the squirrel monkey. ii. response to sinusoidal stimulation and dynamics of peripheral vestibular system. *Journal of neurophysiology*, 34(4):661–675.

[Fisher et al., 2011] Fisher, D. L., Rizzo, M., Caird, J., and Lee, J. D. (2011). *Handbook of driving simulation for engineering, medicine, and psychology*. CRC Press.

[Garone et al., 2017] Garone, E., Di Cairano, S., and Kolmanovsky, I. (2017). Reference and command governors for systems with constraints: A survey on theory and applications. *Automatica*, 75:306–328.

[Gilbert et al., 1995] Gilbert, E. G., Kolmanovsky, I., and Tan, K. T. (1995). Discrete-time reference governors and the nonlinear control of systems with state and control constraints. *International Journal of robust and nonlinear control*, 5(5):487–504.

[Grammatico et al., 2013] Grammatico, S., Blanchini, F., and Caiti, A. (2013). Control-sharing and merging control lyapunov functions. *IEEE Transactions on Automatic Control*, 59(1):107–119.

[Gutman and Cwikel, 1986] Gutman, P. . and Cwikel, M. (1986). Admissible sets and feedback control for discrete-time linear dynamical systems with bounded controls and states. *IEEE Transactions on Automatic Control*, 31(4):373–376.

[Gutman and Cwikel, 1986] Gutman, P.-O. and Cwikel, M. (1986). Admissible sets and feedback control for discrete-time linear dynamical systems with bounded controls and states. *IEEE transactions on Automatic Control*, 31(4):373–376.

[Hartfiel and Stark, 2019] Hartfiel, B. and Stark, R. (2019). Influence of vestibular cues in head-mounted display-based driving simulators. In Kemeny, A., Colombet, F., Merienne, F., and Espié, S., editors, *Proceedings of the Driving Simulation Conference 2019 Europe VR*, pages 25–32, Strasbourg, France. Driving Simulation Association.

[Hogerbrug et al., 2020] Hogerbrug, M., Venrooij, J., Pool, D. M., and Mulder, M. (2020). Simulator sickness ratings reduce with simulator motion when driven through urban environments. In Kemeny, A., Chardonnet, J.-R., and Colombet, F., editors, *Proceedings of the Driving Simulation Conference 2020 Europe VR*, pages 175–178, Antibes, France. Driving Simulation Association.

[Houck et al., 2005] Houck, J. A., Telban, R. J., and Cardullo, F. M. (2005). Motion cueing algorithm development: Human-centered linear and nonlinear approaches.

[Hovd et al., 2014] Hovd, M., Olaru, S., and Bitsoris, G. (2014). Low complexity constraint control using contractive sets. *IFAC Proceedings Volumes*, 47(3):2933–2938.

[Ivleva et al., 2019] Ivleva, V., Holzmann, S., Venrooij, J., and Zachmann, G. (2019). Towards seamless user experiences in driving simulation studies. In Kemeny, A., Colombet, F., Merienne, F., and Espié, S., editors, *Proceedings of the Driving Simulation Conference 2019 Europe VR*, pages 17–24, Strasbourg, France. Driving Simulation Association.

[Kheawhom and Bumroongsri, 2014] Kheawhom, S. and Bumroongsri, P. (2014). Interpolation-based robust constrained model predictive output feedback control. In *22nd Mediterranean Conference on Control and Automation*, pages 396–401. IEEE.

[Kolff et al., 2020] Kolff, M., Venrooij, J., Pool, D. M., and Mulder, M. (2020). Comparison of quality metrics between motion cueing algorithms in a virtual test environment. In Kemeny, A., Chardonnet, J.-R., and Colombet, F., editors, *Proceedings of the Driving Simulation Conference 2020 Europe VR*, pages 93–99, Antibes, France. Driving Simulation Association.

[Koyuncu et al., 2020] Koyuncu, A. B., Erçelik, E., Comulada-Simpson, E., Venrooij, J., Kaboli, M., and Knoll, A. (2020). A novel approach to neural network-based motion cueing algorithm for a driving simulator. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 2118–2125. IEEE.

[La Salle, 1976] La Salle, J. P. (1976). *The stability of dynamical systems*. SIAM.

[Lamprecht et al., 2021] Lamprecht, A., Steffen, D., Nagel, K., Haecker, J., and Graichen, K. (2021). Online model predictive motion cueing with real-time driver prediction. *IEEE Transactions on Intelligent Transportation Systems*.

[Laraba et al., 2017] Laraba, M.-T., Olaru, S., and Niculescu, S.-I. (2017). Linear model predictive control and time-delay implications. *IFAC-PapersOnLine*, 50(1):14406–14411.

[Laraba et al., 2016] Laraba, M.-T., Olaru, S., Niculescu, S.-I., Blanchini, F., Giordano, G., Casagrande, D., and Miani, S. (2016). Guide on set invariance for delay difference equations. *Annual Reviews in Control*, 41:13–23.

[Lazar et al., 2013] Lazar, M., Doban, A. I., and Athanasopoulos, N. (2013). On stability analysis of discrete-time homogeneous dynamics. In *2013 17th International Conference on System Theory, Control and Computing (ICSTCC)*, pages 297–305. IEEE.

[Lee, 2004] Lee, Y. (2004). Receding-horizon control for input constrained linear parameter-varying systems. *IEE Proceedings-Control Theory and Applications*, 151(5):547–553.

[Lee and Kouvaritakis, 2006] Lee, Y. I. and Kouvaritakis, B. (2006). Constrained robust model predictive control based on periodic invariance. *Automatica*, 42(12):2175–2181.

[Limon and Alamo, 2013] Limon, D. and Alamo, T. (2013). Tracking model predictive control. *Encyclopedia of Systems and Control*, pages 1–12.

[Limon et al., 2005] Limon, D., Alvarado, I., Alamo, T., and Camacho, E. (2005). Mpc for tracking of piece-wise constant references for constrained linear systems. *IFAC Proceedings Volumes*, 38(1):135–140.

[Limón et al., 2008] Limón, D., Alvarado, I., Alamo, T., and Camacho, E. F. (2008). Mpc for tracking piecewise constant references for constrained linear systems. *Automatica*, 44(9):2382–2387.

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

[Meiry, 1965] Meiry, J. L. (1965). *The vestibular system and human dynamic space orientation.* PhD thesis, Massachusetts Institute of Technology.

[Mohammadi et al., 2016] Mohammadi, A., Asadi, S., Nelson, K., and Nahavandi, S. (2016). Future reference prediction in model predictive control based driving simulators. In *Australasian conference on robotics and automation (ACRA2016)*.

[Nehaoua et al., 2006] Nehaoua, L., Arioui, H., Espie, S., and Mohellebi, H. (2006). Motion cueing algorithms for small driving simulator. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 3189–3194. IEEE.

[Nguyen et al., 2011a] Nguyen, H., Gutman, P.-O., Olaru, S., Hovd, M., and Colledani, F. (2011a). Improved vertex control for time-varying and uncertain linear discrete-time systems with control and state constraints. In *Proceedings of the 2011 American Control Conference*, pages 4386–4391. IEEE.

[Nguyen, 2014] Nguyen, H.-N. (2014). Constrained control of uncertain, time-varying, discrete-time systems. *Lecture Notes in Control and Infomration Sciences*, 451:17.

[Nguyen et al., 2011b] Nguyen, H.-N., Gutman, P.-O., Olaru, S., and Hovd, M. (2011b). Explicit constraint control based on interpolation techniques for time-varying and uncertain linear discrete-time systems. *IFAC Proceedings Volumes*, 44(1):5741–5746.

[Nguyen et al., 2013] Nguyen, H.-N., Gutman, P.-O., Olaru, S., and Hovd, M. (2013). Implicit improved vertex control for uncertain, time-varying linear discrete-time systems with state and control constraints. *Automatica*, 49(9):2754–2759.

[Nordmark, 1984] Nordmark, S. (1984). *VTI driving simulator: mathematical model of a four-wheeled vehicle for simulation in real time.* Statens Väg-och Trafikinstitut. VTI Rapport 267A.

[Olaru and Dumur, 2005] Olaru, S. and Dumur, D. (2005). Compact explicit mpc with guarantee of feasibility for tracking. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 969–974. IEEE.

[Ormsby, 1974] Ormsby, C. C. (1974). *Model of human dynamic orientation.* PhD thesis, Massachusetts Institute of Technology.

[Ormsby and Young, 1977] Ormsby, C. C. and Young, L. R. (1977). Integration of semicircular canal and otolith information for multisensory orientation stimuli. *Mathematical Biosciences*, 34(1-2):1–21.

[Page, 2000] Page, R. L. (2000). Brief history of flight simulation. *SimTecT 2000 proceedings*, pages 11–17.

[Parduzi et al., 2020] Parduzi, A., Venrooij, J., and Marker, S. (2020). The effect of head-mounted displays on the behavioural validity of driving simulators. In Kemeny, A., Chardonnet, J.-R., and Colombet, F., editors, *Proceedings of the Driving Simulation Conference 2020 Europe VR*, pages 125–132, Antibes, France. Driving Simulation Association.

[Parrish et al., 1975] Parrish, R. V., Dieudonne, J. E., Bowles, R. L., and Martin Jr, D. J. (1975). Coordinated adaptive washout for motion simulators. *Journal of aircraft*, 12(1):44–50.

[Rengifo et al., 2018] Rengifo, C., Chardonnet, J.-R., Paillot, D., Mohellebi, H., and Kemeny, A. (2018). Solving the constrained problem in model predictive control based motion cueing algorithm with a neural network approach.

[Repa and Wierwille, 1976] Repa, B. S. and Wierwille, W. W. (1976). Driver performance in controlling a driving simulator with varying vehicle response characteristics. *SAE Transactions*, pages 2453–2468.

[Reuten et al., 2020] Reuten, A. J. C., Bos, J., and Smeets, J. B. J. (2020). The metrics for measuring motion sickness. In Kemeny, A., Chardonnet, J.-R., and Colombet, F., editors, *Proceedings of the Driving Simulation Conference 2020 Europe VR*, pages 183–186, Antibes, France. Driving Simulation Association.

[Ronellenfitsch et al., 2019] Ronellenfitsch, A., Dong, S., Baumgartner, E., Reuss, H.-C., and Schramm, D. (2019). Objective criteria for motion-cueing evaluation. In Kemeny, A., Colombet, F., Merienne, F., and Espié, S., editors, *Proceedings*

of the Driving Simulation Conference 2019 Europe VR, pages 85–91, Strasbourg, France. Driving Simulation Association.

[Santos, 2018] Santos, T. L. (2018). Modified reference tracking mpc for constrained linear systems. *International Journal of Systems Science*, 49(8):1674–1684.

[Santos et al., 2010] Santos, T. L., Normey-Rico, J. E., and Limón, D. (2010). Explicit input-delay compensation for robustness improvement in mpc. *IFAC Proceedings Volumes*, 43(2):384–389.

[Sato et al., 2019] Sato, T., Takeda, Y., Akamatsu, M., and Kitazaki, S. (2019). Comparison of driver conditions in automated driving systems and transition behaviors in driving simulator versus real proving ground. In Kemeny, A., Colombet, F., Merienne, F., and Espié, S., editors, *Proceedings of the Driving Simulation Conference 2019 Europe VR*, pages 43–50, Strasbourg, France. Driving Simulation Association.

[Schmidt and Conrad, 1970] Schmidt, S. F. and Conrad, B. (1970). *Motion drive signals for piloted flight simulators*, volume 1601. National Aeronautics and Space Administration.

[Scialanga and Ampountolas, 2019] Scialanga, S. and Ampountolas, K. (2019). Interpolating control toolbox (ict). In *2019 18th European Control Conference (ECC)*, pages 2510–2515. IEEE.

[Scibilia et al., 2011] Scibilia, F., Olaru, S., and Hovd, M. (2011). On feasible sets for mpc and their approximations. *Automatica*, 47(1):133–139.

[Soyer et al., 2020a] Soyer, M., Olaru, S., Ampountolas, K., Scialanga, S., and Fang, Z. (2020a). Periodic set invariance as a tool for constrained reference tracking. In *IFAC World Congress*.

[Soyer et al., 2020b] Soyer, M., Olaru, S., and Fang, Z. (2020b). From constraint satisfactions to periodic positive invariance for discrete-time system. In *59th IEEEConference on Decision and Control*.

[Soyer et al., 2020] Soyer, M., Olaru, S., and Fang, Z. (2020). Interpolation based control for reference tracking under constraints. In *2020 European Control Conference (ECC)*, pages 855–860.

[Soyer et al., 2020a] Soyer, M., Olaru, S., and Fang, Z. (2020a). Interpolation-based mca for acceleration rendering. In *Driving Simulation & VR Conference*.

[Soyer et al., 2020b] Soyer, M., Olaru, S., and Fang, Z. (2020b). A novel motion cueing algorithm based on real-time optimization and periodic invariant sets. In *4th IEEE Conference on Control Technology and Applications (CCTA 2020)*.

[Soyer et al., 2021a] Soyer, M., Olaru, S., and Fang, Z. (2021a). Motion cueing control design based on a nonlinear mpc algorithm. *IFAC-PapersOnLine*, 54(6):341–346.

[Soyer et al., 2021b] Soyer, M., Olaru, S., and Fang, Z. (2021b). Mpc delay compensation based on maximal controllable sets for real-time driving simulators. In *2021 25th International Conference on System Theory, Control and Computing (ICSTCC)*, pages 649–654. IEEE.

[Stewart, 1965] Stewart, D. (1965). A platform with six degrees of freedom. *Proceedings of the institution of mechanical engineers*, 180(1):371–386.

[Telban et al., 2000] Telban, R. J., Wu, W., Cardullo, F. M., and Houck, J. A. (2000). Motion cueing algorithm development: Initial investigation and redesign of the algorithms. Technical report.

[Tuchner and Haddad, 2017] Tuchner, A. and Haddad, J. (2017). Vehicle platoon formation using interpolating control: A laboratory experimental analysis. *Transportation Research Part C: Emerging Technologies*, 84:21–47.

[Young and Meiry, 1968] Young, L. R. and Meiry, J. L. (1968). A revised dynamic otolith model. In *Third Symposium on the Role of the Vestibular Organs in Space Exploration, NASA SP-152*, pages 363–368.

[Zöller et al., 2019] Zöller, C., Müller, A., Eggert, L., Winner, H., and Abendroth, B. (2019). Applicability of head-mounted displays in driving simulation. In Kemeny, A., Colombet, F., Merienne, F., and Espié, S., editors, *Proceedings of the Driving Simulation Conference 2019 Europe VR*, pages 9–15, Strasbourg, France. Driving Simulation Association.

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre: Amélioration de la restitution des stimuli sensoriels sur simulateur de conduite
Mots clés: Commande sous contraintes, simulation de conduite, retard, prediction, optimisation

Résumé: Les simulateurs dynamiques de conduite à haute performance restituent les accélérations du véhicule à l'aide d'un système de mise en mouvement de plateforme. Le conducteur est immergé dans un environnement de synthèse multi-sensoriel (stimulations visuelles, haptiques, vestibulaires et sonores). La performance d'un tel simulateur en dehors de sa capacité mécanique (latence visuelle, le retard de restitution de mouvement, etc...) et de la cohérence temporelle des différents stimuli sont des facteurs pouvant conditionner la qualité de la perception.
La présente thèse a pour objectif de développer un algorithme de restitution de mouvement qui minimise le temps de calcul et corrige le retard du simulateur tout en conservant la performance de restitution de mouvement. Cet algorithme de restitution fait l'objet d'un intérêt spécifique de la communauté de la simulation de conduite via la dénomination MCA (Motion Cueing Algorithm). Cette thèse porte sur la commande optimale et en particulier sur la récente utilisation de la commande prédictive par modèle (ou Model Predictive Control ou MPC) comme base de l'algorithme. Les différentes pistes d'amélioration évoquées dans cette thèse concernent l'utilisation des ensembles invariants pour la poursuite de trajectoire en temps réel.
Ainsi, un algorithme de compensation de retard inertiel est proposé dans une version allégée du point de vue du temps d'exécution. Un algorithme non prédictif basée sur la commande par interpolation adapté à la poursuite de trajectoire dynamique est également proposé. Enfin, une base théorique permettant la relaxation des concepts d'invariance est conçue afin de minimiser la complexité des problèmes d'optimisation utilisés dans la commande prédictive.

Title: Improvement of the sensory stimuli restitution on driving simulator
Keywords: Constrained control, driving simulation, delay, prediction, optimization

Abstract: High performance driving simulators reproduce vehicle acceleration based on adequate motion systems. The driver is immersed in a multi-sensorial (visual, haptic, vestibular and sound stimulation) environment of synthesis. The performance of driving simulator outside of its mechanical capacity (the visual latency, the delay of restitution of movement, etc...) and of the temporal coherence of the different stimuli are factors that can condition the validity of the perception.
The present thesis aims at to develop a motion restitution algorithm that minimizes the calculation time and corrects the simulator delay while maintaining the performance of motion restitution. This restitution algorithm receives a specific interest from the driving simulation community through the denomination MCA (Motion Cueing Algorithm). This thesis work deals with optimal control and particularly on the recent use of Model Predictive Control (MPC) as a base of the algorithm. The different ways for improving concern the set invariance use in the control design for the trajectory tracking.
Thus, a delay compensation algorithm is proposed with a less computational burden. A non predictive algorithm is also designed based on the interpolation-based control technique adapted to the dynamic trajectory tracking. Finally, new set theoretic notions relaxing set invariance notions are proposed in order to minimize the complexity of optimization problems in the MPC procedure.

Maison du doctorat de Université Paris-Saclay
2e étage, aile ouest, École normale supérieure Paris-Saclay
4 avenue des Sciencs
91190 Gif-sur-Yvette, France