# Optimization and control of nonholonomic vehicles and vehicles formations

Amélia Cristina Duque Caldeira Matos

PDMA - Programa de Doutoramento em Matemática Aplicada
Universidade do Porto

2011

# Optimization and control of nonholonomic vehicles and vehicles formations

**Amélia Cristina Duque Caldeira Matos**

Trabalho efectuado sob a orientação do
Professor Doutor **Fernando Arménio da Costa Castro e Fontes**
e da
Professora Doutora **Dalila Benedita Machado Martins Fontes**

PDMA - Programa de Doutoramento em Matemática Aplicada
Universidade do Porto

2011

Dissertação apresentada à
Faculdade de Ciências da Universidade do Porto
para a obtenção do grau de Doutor em Matemática Aplicada

# Contents

# List of Figures

# List of Tables

# Abstract

**Optimization and control of nonholonomic vehicles and vehicles formations**

Control theory, in this thesis, is concerned with dynamic systems and their optimization over time. We work in applications of optimization techniques based on control theory. The theory is applied to control nonholonomic vehicles, in particular, wheeled mobile robots, using optimization-based techniques, namely, Optimal Control and Model Predictive Control.

An key motivation for this research topic stems from the fact that nonholonomic systems pose considerable challenges to control system designers since those systems cannot be stabilized by smooth, time-invariant, state-feedback control laws.

Model Predictive Control is a technique that constructs a feedback law by solving on-line a sequence of open-loop optimal control problems, each of them using the currently measured state of the plant as its initial state.

Similarly to Optimal Control, Model Predictive Control has an inherent ability to deal naturally with constraints on the inputs and on the state. Since the controls are obtained by optimizing some criterion, the method possesses some desirable performance properties, and also intrinsic robustness properties. These facts can partially explain the substantial impact it has made on industry with thousands of applications reported.

There has been an intense research addressing a wide range of issues such as stability, robustness, performance analysis and state estimation. In the recent years, there has been research interest in developing MPC for nonholonomic systems, in particular wheeled mobile robots.

Moreover, since nonholonomic systems cannot be stabilized by a time-invariant continuous feedback, some care is required when studying the trajectories resulting from MPC controllers that allow discontinuous feedbacks. Nonholonomic systems are, therefore, an important motivation to develop methodologies that allow the construction of discontinuous feedback controls.

This thesis also addresses formation of mobile robots. In this respect, we study the control to maintain a given formation and also aspects related to the reorganization of the formation. Once again, optimization is a key ingredient in both the control and reorganization of the formations.

**Keywords:** Nonholonomic systems, Optimal Control, Model Predictive Control, formation of mobile robots.

# Sumário


**Optimização e controlo de veículos não holonómicos e formação de veículos**

A teoria de controlo usada neste trabalho lida com sistemas dinâmicos e a sua optimização ao longo do tempo. As técnicas de optimização são aplicadas na teoria de controlo, nomeadamente, no controlo de veículos não holonómicos (em particular, robôs móveis) utilizando técnicas de Controlo Óptimo e Controlo Preditivo.

Uma das grandes motivações desta investigação decorre do facto de que os sistemas não holonómicos colocam desafios consideráveis, uma vez que estes sistemas não podem ser estabilizados por realimentações contínuas.

De forma semelhante ao Controlo Óptimo, o Controlo Preditivo tem a capacidade inerente de lidar naturalmente com restrições de entradas e de estado. Uma vez que os controlos são obtidos por meio da optimização de algum critério, o método possui algumas propriedades de desempenho desejáveis, e também propriedades de robustez intrínseca. Estes factos podem explicar em parte o impacto substancial que fez na indústria com milhares de aplicações relatadas.

Tem havido uma intensa pesquisa abordando uma ampla gama de questões como estabilidade, robustez e análise de desempenho. Nos últimos anos, tem havido um interesse em desenvolver Controlo Preditivo para sistemas não holonómicos, em particular, robôs móveis.

Além disso, é conhecida a impossibilidade de estabilizar sistemas não holonómicos recorrendo a leis de controlo suaves e invariantes no tempo, logo, alguns cuidados são necessários quando se estudam as trajectórias resultantes de controladores MPC que permitem realimentações descontínuas. Sistemas não holonómicos são, portanto, uma motivação importante para desenvolver metodologias que permitam a construção de controlos de realimentação descontínua.

Esta tese aborda também a formação de robôs móveis. Neste campo estudamos o controlo para manter uma dada formação e também os aspectos relacionados com a reorganização da formação. Mais uma vez, a optimização é um ingrediente - chave quer no controlo quer na reorganização das formações de robôs.



**Keywords:** Sistemas não holonómicos, Controlo Óptimo, Controlo Preditivo, formação de robôs móveis.

# Acknowledgements

Throughout this journey many people took part in this work, either as friends, teachers or colleagues. Here is my affection for their contribution in this work.

I want to thank in a very special and personal way to my advisors Professor Fernando Fontes and Professor Dalila Fontes for their good will and dedication to guide my work. Their patience and knowledge coupled with their human qualities made the realization of this thesis a pleasure. They accepted my work rhythm and believed that this thesis would take place, manifesting me all their availability with immediate responses when necessary, supporting and encouraging me at all time. Without them this work would not have been achieved. Thank you!

I would like to thank Professor Lobo Pereira and Professor António Paulo Moreira for our discussions during the PDMA Symposiums and for giving me useful comments and advices on my work.

Thanks also to Professor Martins de Carvalho for the didactic lunches on Fridays.

Many thanks to the people of Institute for Systems and Robotics – Porto (ISR-P) at the Faculdade de Engenharia da Universidade do Porto (FEUP) for providing a warm and friendly working environment. I would like to thank in particular to Maria do Rosário de Pinho and Margarida Ferreira for their friendship, affection and for making me feel part of the team. Special thanks to Sofia Lopes for her support, friendship and words of encouragement.

A special word to my colleagues of Instituto Superior de Engenharia do Porto (ISEP) who were able to create a good atmosphere which led to the development of this study. A special thanks to my colleague and friend Isabel Figueiredo.

Fortunately, I am a person who has many good friends. Many thanks also to the rest of my friends who are too many to name. You are great!

My power supply is my family and I have an amazing and big family. Their support has been unconditional all these years, and supported me whenever I needed it.

The attention and support, wisdom, intelligence, love and affection of my mother and father, Inês and Joaquim, have always been a comforting presence throughout my entire life. My special love to a very strong woman, my mother, for the times she has replaced me in my role of mother during this work. I want to mention especially a sweet man, my father, who unfortunately died very recently and whose support was ever present - he was the pillar of family. By his smile and

good humor, and for being a great father and a great grandfather, is to him that this thesis is dedicated.

My acknowledgements would be incomplete without mention my sister Carla, my brothers, Paulo and Hugo, and my nephews, Tomás, Dinis and Guilherme.

Thanks also to D$^a$ Lininha, Sr. Matos, Dulce, Joana and Susana.

Last but not least, I wish to express my special love to my amazing husband, António, for having unlimited patience and constant faith in me, for his love, encouragement and understanding during the course of this work. I thank also our lovely three children, Inês, Gonçalo and Alexandre, who have been always good kids. Their intelligence, their smile and their enthusiasm for learning, inspired me to overcome one difficulty after another.

Once again, thank you all!

**Obrigada a todos!**

## Dedication

To my father, Joaquim Marques Caldeira,
1939-2011
He was always present in my life. For the years of love, encouragement and support.

# Chapter 1

# Introduction

## 1.1 Motivation and scope

This Thesis addresses the control of nonholonomic vehicles, in particular, wheeled mobile robots, using optimization-based techniques, namely, Optimal Control and Model Predictive Control. It also addresses formation of mobile robots. In this respect, we study the control to maintain a given formation and also aspects related to the reorganization of the formation. Once again, optimization is a key ingredient in both the control and reorganization of the formations.

The application area of mobile robots is large and still growing. Some make life easier for humans, like automatic vacuum cleaners, post delivery robots in office buildings or order-pick robots in automated warehouses, some doing work that otherwise would be very dangerous or even impossible due to hazardous or unreachable environments, like firefighting, rescue or spy missions, namely, for hostage situations, exploration of deep oceans or extraterrestrial planet environments.

Nonholonomic systems most commonly arise in finite dimensional mechanical systems where the constraints that are imposed on the motion are not integrable. That is, the constraints cannot be written as time derivatives of some function of the generalized coordinates. Such constraints can usually be expressed in terms of non integrable linear velocity relationships. Nonholonomic control systems result from formulations of nonholonomic systems that include control inputs. This class of nonlinear control systems has been studied by many researchers, and the published literature is now extensive. See, for example, [KM95] and the references therein, [MLS94], [SSVO09].

The interest in these nonlinear control problems is motivated by the fact that these problems are not amenable to methods of linear control theory, and they are not transformable into linear control problems in any meaningful way. Hence, these are nonlinear control problems that require fundamentally nonlinear approaches. On the other hand, these nonlinear control problems are sufficiently special that good progress can be made. Nonholonomic systems are typically completely controllable but instantaneously they cannot move in certain directions. Although these systems are allowed to move, eventually, in any direction, at a certain time or

state there are constraints imposed on the motion – the so-called nonholonomic constraints. Some of the interesting examples are the wheeled vehicles, which, at a certain instant can only move in a direction perpendicular to the axle connecting the wheels.

Optimization-based methods plays an important role in control engineering, in particular, if constraint handling or efficiency is concerned. Typical control problems are the optimization of the time spent or of the energy consumed.

We address the design of controllers for nonlinear systems. Among the methodologies well suited to deal with this problem we concentrate on the optimization-based ones, namely, open-loop Optimal Control methods and Model Predictive Control.

Model Predictive Control is an increasingly popular control technique that has been developed both by the systems theory community (where it is also known as Receding Horizon Control), and by the process engineering community (where it is often referred to by commercial names such as Dynamic Matrix Control). Originally developed to cope with the control needs of petroleum refineries, it is currently successfully used in a wide range of applications, not only in the process industry but also other processes ranging from automotive to clinical anesthesia. This technique constructs a feedback law by solving on-line a sequence of open-loop optimal control problems, each of them using the currently measured state of the plant as its initial state.

Similarly to optimal control, MPC has an inherent ability to deal naturally with constraints on the inputs and on the state. Since the controls are obtained by optimizing some criterion, the method possesses some desirable performance properties, and also intrinsic robustness properties [MS97]. These facts can partially explain the substantial impact it has made on industry ([QB97, QB98]).

Several recent publications provide a good introduction to theoretical and practical issues associated with MPC technology (see for example, the books [Mac02], [CB04] and the survey papers ([MRRS00], [QB97],[FA03]).

Moreover, since nonholonomic systems cannot be stabilized by a time-invariant continuous feedback, some care is required when studying the trajectories resulting from MPC controllers that allow discontinuous feedbacks. Nonholonomic systems are, therefore, an important motivation to develop methodologies that allow the construction of discontinuous feedback controls.

Formation control has become one of the well-known problems in multi-robot systems. Research in coordination and control of teams of several vehicles (that may be robots, ground, air or underwater vehicles) has been growing fast in the past few years. Application areas include unmanned aerial vehicles (UAVs) [BPA93, WCS96], autonomous underwater vehicles (AUVs) [SHL01], automated highway systems (AHSs) [Ben91, SH99] and mobile robotics [Yam99, YAB01]. While each of these application areas poses its own unique challenges, several common threads can be found. In most cases, the vehicles are coupled through the task they are trying to accomplish, but are otherwise dynamically decoupled, meaning the motion of one does not directly affect the others. For a survey in cooperative control of multiple vehicles systems, see for example the work by Murray [Mur07].

The problem of determining how a structured formation of robots can be reorganized into


Figure 1.1: a) Birds flying in a formation; b) Airplanes in a formation flight (photos taken from the internet).

another formation is studided in this work. Possible applications can be found in surveillance, damage assessment, chemical or biological monitoring, among others, where the switching to another formation, not necessarily predefined, may be required due to changes in the environment. Among the possible actions to reorganize the formation into a new desired geometry, we study the ones that optimize a pre-determined performance measure.

## 1.2 Main contributions

The ideas that set the scene in which the research work is made, are reported first. We provide an overview on the field of nonholonomic system control. During the presentation, theoretical problems are discussed and practical applications described. In the recent years, there has been research interest in developing MPC for nonholonomic systems, in particular wheeled mobile robots. See [VEN01, Fon01, Fon02b, FM03a, GH05, GH06]. Then we take a trip to two particularly optimization-based techniques: Optimal Control and the Model Predictive Control. After the optimal control problem is overviewed and discussed, we prepare the scenario to introduce Model Predictive Control as a method of generating closed-loop control laws by solving a sequence of open-loop optimal control problems. All of this served as a motto for the next chapters.

The continuity of the controls resulting from the open-loop optimal control problems was a common assumption in most of the earliest approaches to MPC. This assumption, in addition to being very difficult to verify, was a major obstacle in enabling MPC to address a broader class of nonlinear systems. Already in the eighties of last century this problem has been reported in [SS80] and [Bro83], saying that, some nonlinear systems cannot be stabilized by a continuous

feedback. In [Fon01] the assumption of all previous continuous-time MPC schemes was relaxed: the continuity of the controls solving the open-loop optimal control problems as well as the continuity of the resulting feedback laws. So, we have been interested in MPC frameworks for nonholonomic systems, such as wheeled vehicles, which require discontinuous feedbacks to be stabilized. The control functions are computed via an optimisation algorithm, hence in general these functions need to be described by a finite number of parameters (the so called finite parameterizations of the control functions). In this thesis, we report results on the implementation of a stable MPC strategy for a wheeled robot, where finite parameterizations for discontinuous control functions are used, resulting in efficient computation of the control strategies.

One of the contributions of this work consists in the control problem for nonholonomic wheeled mobile robots moving on the plane, and in particular the use of feedback techniques for achieving a given motion task, our analysis is to a case of a robot workspace free of obstacles. We implemented a version of a predictive control algorithm based on the structures described in [Fon01, Fon02a, FMG07]]: a stabilizing MPC strategy to a wheeled robot and established conditions under which steering to a set is guaranteed. We also derived a set of design parameters satisfying the conditions for the control of a unicycle mobile robot.

When considering path-following instead of trajectory-tracking, the degree of freedom on how fast we move along the path can be used to increase performance in certain systems [AHK05]. MPC can explore effectively this degree of freedom and in addition it can deal naturally with the constraints omnipresent in path-following problems. Then, taking advantage of such considerations, we also implemented a version of a predictive control algorithm based on the structures described in [AHK05]: if the path-following problem is converted into a trajectory MPC framework will find a feedback control to follow the path given.

Formations of undistinguishable agents arise frequently both in nature and in mobile robotics. Another contribution of this work results in an application of Model Predictive Control technique to control of vehicles in a formation. Dealing with multiagent coordination, incorporating also the trajectory control component that allows maintaining or changing the formation while following a specified path in order to perform cooperative tasks. Although several optimization problems have to be solved, the control strategy proposed results in a simple and efficient implementation where no optimization problem needs to be solved in real-time at each vehicle.

Continuing our journey in the vehicle formation, we treat in this thesis thesis the problem of dynamically switching the geometry of a formation of a number of undistinguishable vehicles. We proposed methodology that is very flexible, in the sense that it easily allows for the inclusion of additional problem features, e.g. imposing geometric constraints on each agent or on the formation as a whole, using nonlinear trajectories, among others. The optimization algorithm that have been developed in this work decides how to reorganize a formation of vehicles into another formation of different shape with collision avoidance and vehicle traveling velocity choice, where each vehicle can also modify its path by changing its curvature (for example to avoid obstacles). This is a relevant problem in cooperative control applications. The method proposed

here should be seen as a component of a framework for multivehicle coordination/cooperation, which must necessarily include other components such as a trajectory control component.

## 1.3 Thesis overview

The introductory chapters (2, 3 and 4) set the background for the main subject of the thesis: Nonholonomic Systems, Optimal Control and Model Predictive Control. Theses chapters serves as the support for the work developed, they do not contain original results. The remaining chapters provided the original contribution of the thesis.

A brief outline of the content of the various chapters is presented next:

**Chapter 2:** In this chapter, we study the effect of nonholonomic constraints on the behavior of a robotic system. The Chapter starts with several notions from differential geometry and by reviewing the key concepts of nonholonomic systems. Then, a brief study on accessibility, controllability and stability is presented. The stabillity and controllability are two fundamental concepts in control theory: stability usualy involves feedback and controllability assesses whether an action trajectory exists that leads to a specified goal state. In the end, the classification of the possible motions tasks: point stabilization, trajectory tracking and path-following is made.

The models of the systems that are used in this work to illustrate the application of the controls techniques are introduced here.

**Chapter 3:** The chapter starts with a brief historical account of the theory of Optimal Control. Then the optimal control problem is defined and some definitions that are related to this problem (model, performance criterion and constraints) is reviewed. A version of the Maximum Principle is also provided in this chapter . At the end the special case when the system dynamics are linear and the cost is quadratic is focused.

**Chapter 4:** This chapter introduces the basic principle of Model Predictive Control. This method is introduced as a process of generating closed-loop control laws by solving a sequence of open-loop optimal control problems. An algorithm, based on the MPC technique, that generate stabilizing feedback control for a universal class of nonlinear time-varying systems: the class of the open-loop uniformly asymptotically controllable systems is given here and will be used throughout the dissertation.

We adress two control schemes where stability is assured: Control Lyapunov Functions MPC and Unconstrained MPC schemes. The first class of schemes use a control Lyapunov function and in [Fon01] it was guaranteed the stability of the resultant closed loop system, by choosing the design parameters to satisfy a certain stability condition. The second class of schemes uses a controllability assumption in terms of the stage cost instead. The

framework of [RA11] extended the results on Model Predictive Control without terminal constraints and terminal cost functions from the discrete-time case, to continuous-time.

**Chapter 5:** In Chapter 5, we investigate MPC frameworks for nonholonomic systems, such as wheeled vehicles, which require discontinuous feedbacks to be stabilized. The control functions are computed via an optimisation algorithm, hence in general these functions need to be described by a finite number of parameters (finite parameterizations of the control functions). In this Chapter, we report results on the implementation of a stable MPC strategy for a wheeled robot, where finite parameterizations for discontinuous control functions are used, resulting in efficient computation of the control strategies.

**Chapter 6:** In this chapter, we distinguish the problems of (i) point stabilization, (ii) trajectory tracking, and (iii) path-following. In point stabilization we aim to drive the state (or position) of our system to a pre-specified point (usually the origin is considered, without loss of generality). A parking manoeuver is an example of such application in wheeled vehicles. In trajectory tracking we start at a given initial configuration and aim to follow as close as possible a trajectory in state space (i.e., a geometric path in the cartesian space together with an associated timing law). In path-following we just aim that the robot position follows a geometric path in the cartesian space, but no associated timing law is specified.

In this Chapter results on the implementation of a stabilizing MPC strategy to a wheeled robot.is reported. Conditions under which steering to a set is guaranteed are established. A set of design parameters satisfying all these conditions for the control of a unicycle mobile robot are derived.

We also discuss the use of MPC to address the problem of path-following of nonholonomic systems. We argue that MPC can solve this problem in a effective and relatively easy way, and has several advantages relative to alternative approaches. We address the path-following problem by converting it into a trajectory-tracking problem and determine the speed profile at which the path is followed inside the optimization problems solved in the MPC algorithm. The MPC framework solves a sequence of optimization problems that find an initial point, a speed profile, and a feedback control to track the trajectory of a virtual reference vehicle, i.e., the MPC framework will find a feedback control to follow the path given.

**Chapter 7:** In this chapter we propose a control scheme for a set of vehicles moving in a formation. The control methodology selected is a two-layer control scheme where each layer is based on MPC. The first layer, the trajectory controller, is a nonlinear controller since most vehicles are nonholonomic systems and require a nonlinear, even discontinuous, feedback to stabilize them. The trajectory controller, a model predictive controller, computes control law and only a small set of parameters needs to be transmitted to each vehicle at each iteration. The second layer, the formation controller, aims to compensate for small changes around a nominal trajectory maintaining the relative positions between vehicles.

We argue that the formation control can be, in most cases, adequately carried out by a linear model predictive controller accommodating input and state constraints. This has the advantage that the control laws for each vehicle are simple piecewise affine feedback laws that can be pre-computed off-line and implemented in a distributed way in each vehicle.

**Chapter 8:** The problem of switching the geometry of a formation of undistinguishable vehicles by minimizing some performance criterion is studied in this chapter. We have developed an optimization algorithm to decide how to reorganize a formation of vehicles into another formation of different shape with collision avoidance and vehicle traveling velocity choice. Moreover, each vehicle can also modify its path by changing its curvature. The formation switching performance is given by the time required for all vehicles to reach their new position, which is given by the maximum traveling time amongst individual vehicle traveling times. Since we want to minimize the time required for all vehicles to reach their new position, we have to solve a minmax problem. However, our methodology can be used with any separable performance function. The algorithm proposed is based on a dynamic programming approach.

**Chapter 9:** Here we conclude this thesis by providing a summary of contributions and posing some related open questions to motivate further research.

# Chapter 2

# Nonholonomic systems

## 2.1 Introduction

Nonlinear control is one of the biggest challenges in modern control theory. While linear control system theory has been well developed, it is the nonlinear control problems that present the greatest difficulties. They have gained importance in many industrial areas and research has had significant developments in past years.

Nonholonomic systems are a class of nonlinear systems frequently appearing in robotics (for example, robot manipulators, mobile robots, wheeled vehicles and underwater robots). In the last few years, the control of these kind of systems has been the subject of considerable research effort. The design of stabilizing feedback controllers for these systems offers some interesting challenges. Namely, these systems are inherently nonlinear: they cannot be handled by any linear control method and are not transformable into linear systems (even locally) in any meaningful way [KM95]. Furthermore, a (time-invariant) feedback law capable of stabilizing this class of systems must be allowed to be discontinuous [Bro83]. As a consequence, nonclassical definitions of a solution to a differential equation are needed to analyze the resulting trajectories.

Nonholonomic systems are, typically, completely controllable but instantaneously they cannot move in certain directions. Although these systems are allowed to reach any point in the state space and to move, eventually, in any direction, at a certain time or state there are constraints imposed on the motion (nonholonomic constraints).

To perform locomotion in mobile robots the use of wheels is the most common mechanism. Wheeled mobile robots cannot move sideways due to the rolling without slipping constraint of the wheels, at a certain instant they only can move in a direction perpendicular to the axle connecting the wheels. The problem of putting a wheeled vehicle in the same orientation but some distance from its current position, has become a well-known problem. Consider, e.g., a vehicle whose dynamics we are familiar with: a car. Consider the car performing a parking maneuver (see Figure 2.1). This type of behavior is a special case of nonholonomic behavior. From the everyday experience with wheeled vehicles, it can be derived that this class

is controllable, but to prove this a characterization of nonholonomic systems is needed and will be given in the next section.

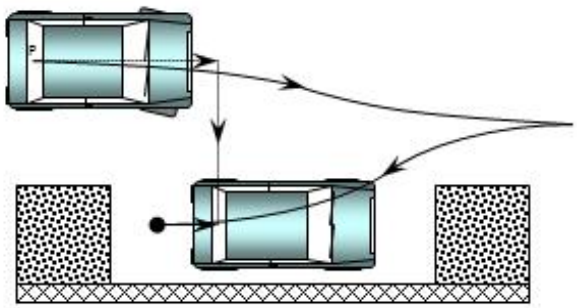

Figure 2.1: Car in a parking manoeuver: cannot move sideways.

In mechanics, the kinematic constraints (i.e., system constraints whose expression involves generalized coordinates and velocities) are usually encountered in the Pfaffian form [MLS94]. The goal in this Chapter is to provide tools for analyzing and controlling nonholonomic mechanical systems.

The mathematical approach to this type of problem can be done through tools of differential geometry. Systematic development of the theory was initiated more than 150 years, based on a series of classic articles of mathematicians and physicists such as Chaplygin, Carathéodory, Hertz, among others. Despite this, only recently the study of control problems for such systems has bigen. Brockett, Isidori, Sussmann and others in the 1970s introduced the methods of differential geometry in the context of nonlinear control. Since then, the development of theoretical tools for design of control laws for a large class of nonlinear control systems has been done. In 1995, Kolmanovsky and McClamroch, said "The published literature on control of nonholonomic systems has grown enormously during the last six years..." [KM95], so, it has been only about 20 years ago that the study of control problems for nonholomomic systems had his "Big Bang".

The nonholonomic systems are a typical class of systems with strong nonlinear nature. As pointed out in a famous paper of Brockett [Bro83], nonholonomic systems cannot be stabilized by continuously differentiable, time-invariant, state feedback control laws. To overcome the limitations imposed by Brockett's result, a great number of approaches have been proposed for the stabilization of nonholonomic control systems. The work of Kolmanovsky and McClamroch (see [KM95] and the references therein) serves as a tutorial presentation for many of the developments in the control of nonholonomic systems. They presented a clear and accessible stages of development of the theory stating models, techniques of control in open loop and closed methods of trajectory planning.

The book by Murray, Li, and Sastry [MLS94] provides a general introduction to nonholo-

nomic control systems and present the study of nonholonomic constraints on the behavior of robotic systems. Another interesting book and more recent (2009) is the book by Siciliano, Sciavicco, Villani and Oriolo [SSVO09], where the concepts are introduced in a coherent and didactic way. Part of this book deal with wheeled mobile robots. It provides techiques for modelling, planning and control wheeled mobile robots. It aims to describe the basic concepts of the theory of nonholonomic systems, especially mobile robots with Pfaffians constraints. To do this, it need to state the basic concepts of differential geometry applied to holonomic and nonholonomic systems, leading to the recognition of nonholonomic systems and the verification of their controllability.

The Kinematic constraints can be seen as a control system evolving on a manifold. And, in turn, a control system is a family of vector fields parameterized by the controls. In this scenario, differential geometry comes into play, since the qualitative properties of a control system depend on the properties of vector fields and interactions between them. The basic tool to understand such interactions is their Lie bracket.

The first part of this Chapter recalls several notions from differential geometry and starts by reviewing the key concepts of nonholonomic systems. Then, it is presented a brief study on accessibility, controllability and stability. The stabillity and controllability are two fundamental concepts in control theory: stability usualy involves feedback and controllability assesses whether an action trajectory exists that leads to a specified goal state. In the end, the classification of the possible motion tasks: point stabilization, trajectory tracking and path-following is made.

The models of the systems that will be used in this work to illustrate the applications of the controls techniques, will be introduced in this Chapter.

## 2.2 On nonholonomic systems

Consider a mechanical system whose configuration can be described by a vector of generalized coordinates $\mathbf{q} = (q_1, q_2, ..., q_n)^T \in Q$. The $n$-dimensional configuration space $Q$ (i.e., the space of all possible robot configurations) is a smooth manifold, locally diffeomorphic[1] to $\mathbb{R}^n$.

The generalized velocity at a generic point of a trajectory $\mathbf{q}(t) \in Q$ is a vector $\dot{\mathbf{q}} = (\dot{q}_1, \dot{q}_2, ..., \dot{q}_n)^T$ belonging to the tangent space $T_q(Q)$.

The movement of mechanical systems, in many cases, is submitted to certain constrains which are permanente enforced during the movement and taking the form of algebraic relations between positions and velocities of particular points of the system. So, a constraint on a mechanical system restricts the motion of the system by limiting the set of paths which the system can follow.

[1] A bijective map $f : M \to N$ between two manifolds $M$ and $N$ is a *diffeomorphism* if $f$ and $f^{-1}$ are differentiable. $f$ is called $C^n$ *diffeomorphism* if it is $n$ times continuously differentiable. $M$ and $N$ are *diffeomorphic* if there exists a diffeomorphism $f : M \to N$ and $M$ and $N$ are $C^n$ *diffeomorphic* if there is an $n$ times continuously differentiable bijective map $f : M \to N$, whose inverse is also $n$ times continuously differentiable.

Two distinct types of restrictions may well be observed: geometric constraints and kinematic constraints.

**Geometric constraints** : may exist or be imposed on the mechanical system, they are represented by analytical relations between the generalized coordinates $\mathbf{q}$ of a mechanical system:

$$h_i(\mathbf{q}) = 0 \qquad i = 1, ..., l \quad < n. \tag{2.1}$$

Each $h_i$ is a mapping from $Q$ to $\mathbb{R}$ which restricts the motion of the system. When the system is subjected to $l$ geometric independent constraints then, if the implicit function theorem conditions are satisfied [2], $l$ generalized coordinates can be eliminated and $n - l$ coordinates are sufficient to provide a full description of the system configuration. Assuming that the constraints are linearly independent and hence the matrix

$$\frac{\partial h}{\partial \mathbf{q}} = \begin{bmatrix} \frac{\partial h_1}{\partial \mathbf{q}_1} & \cdots & \frac{\partial h_1}{\partial \mathbf{q}_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_k}{\partial \mathbf{q}_1} & \cdots & \frac{\partial h_k}{\partial \mathbf{q}_n} \end{bmatrix}$$

is full rank (i.e., $\det\left(\frac{\partial h}{\partial \mathbf{q}}\right) \neq 0$). In general the use of this procedure is only valid locally, and may introduce singularities. A convenient alternative is to "eliminate" the constraints by choosing a set of coordinates. These new coordinates parameterize all allowable motions of the system and are not subject to any further constraints. Examples include obstacles in the path.

**Kinematic constraints** : they are represented by analytical relations between the coordinates $\mathbf{q}$ and velocities $\dot{\mathbf{q}}$ :

$$a_i(\mathbf{q}, \dot{\mathbf{q}}) = 0, \qquad i = 1, ..., k < n \tag{2.3}$$

and constrain the instantaneous admissible motion of the mechanical system by reducing the set of generalized velocities that can be attained at each configuration.

---

[2]

**Theorem 1** ***(Implicit function theorem)*** *Let $A \subset \mathbb{R}^n \times \mathbb{R}^m$, be an open subset and $F : A \to \mathbb{R}^m$ a function of class $C^1$. Let $(\mathbf{a}, \mathbf{b}) \in A$ and consider the system of equations*

$$F(\mathbf{x}, \mathbf{y}) = F(\mathbf{a}, \mathbf{b}). \tag{2.2}$$

*Assume that the $(m \times n)$ matrix of partial derivates*

$$B = \left(\frac{\partial F_i}{\partial y_k}(\mathbf{a}, \mathbf{b})\right)$$

*is invertible ($\det(B) \neq 0$). Then there exists an open subset $U \subset \mathbb{R}^n$ containing $\mathbf{a}$ and an open subset $V \subset \mathbb{R}^m$ containing $\mathbf{b}$ such that $U \times V \subset A$ and such that, for all $\mathbf{x} \in U$, there exists a unique $\mathbf{y} = f(\mathbf{x}) \in V$ such that $(\mathbf{x}, \mathbf{y})$ is a solution to the system of equations (2.2).*

Kinematic constraints are classically divided into two classes: holonomic constraints and nonholonomic constraints.

**Definition 1** *[MLS94]*
*- constraints that can be integrated to yield constraints on the position variables, i.e., the constraint can be reduced to the form of expression* (2.1) *and they are called* ***holonomic constraints****;*
*- constraints for which this integration is not possible, called* ***nonholonomic constraints****, and the system that has these restrictions is called* ***nonholonomic system****.*

If the system is nonholonomic, the number of degrees of freedom (or the number of independent velocity) is equal to the number of independent generalized coordinates minus the number of nonholonomic constraints.

In mechanics, permissible movements of the system are usually limited by constraints of the form [MLS94],

$$a_i^T(\mathbf{q})\,\dot{\mathbf{q}} = 0, \quad i = 1, ..., k < n, \quad \text{or, in a matrix form:} \quad A^T(\mathbf{q})\ \dot{\mathbf{q}} = 0, \tag{2.4}$$

where $A^T \in \mathbb{R}^{k\times n}$ represents a set of $k$ velocities constraints. A restriction like this is called a *Pfaffian restriction.* The vectors

$$a_i : Q \to \mathbb{R}^n, \quad i = 1, ..., k < n,$$

are assumed to be smooth and linearly independent, so $A^T(\mathbf{q})$ is full row rank at $\mathbf{q} \in Q$.

The existence of $l$ holonomic constraints, as expressed in (2.1), imply the existence of $l$ kinematic constraints, since

$$\frac{dh_i(\mathbf{q}(t))}{dt} = \frac{\partial h_i \mathbf{q}(t)}{\partial \mathbf{q}}\ \dot{\mathbf{q}} = 0, \qquad i = 1, ...l. \tag{2.5}$$

However, the converse is not necessarily true: it may occur that the kinematics constraints (2.4) are not integrable, i.e., cannot be put in the form (2.1). The constraints and the system are nonholonomic.

The presence of nonholonomic constraints limits the system mobility in a completely different way when compared to holonomic constraints.

The determination whether a system is holonomic or not is not an easy task. Consider the case in which there is a single speed constraint. This can be illustrated, considering a single Pfaffian constraint

$$a^T(\mathbf{q})\ \dot{\mathbf{q}} = \sum_{j=1}^{n} a^j(\mathbf{q})\ \dot{q}_j = 0. \tag{2.6}$$

Once a Pfaffian constraint restricts the allowable velocities of the system but not necessarily the configurations, in general, it cannot be represented as an algebraic constraint on the configuration space. For this constraint to be integrable [SSVO09], there must exist a scalar function $h : Q \to \mathbb{R}^k$ and an integrating factor $\gamma(\mathbf{q}) \neq 0$ such that

$$\gamma(\mathbf{q})\, a^j(\mathbf{q}) = \frac{\partial h}{\partial \mathbf{q}_j}(\mathbf{q}), \qquad j = 1, ..., n. \tag{2.7}$$

The converse also holds: if there exists a $\gamma(\mathbf{q}) \neq 0$ such that $\gamma(\mathbf{q})a(\mathbf{q})$ is the gradient vector of some scalar function $h(\mathbf{q})$, then constraint (2.6) is integrable. By using again [SSVO09], the integrability condition (2.7) may be replaced by

$$\frac{\partial\left(\gamma a^k\right)}{\partial q_j} = \frac{\partial\left(\gamma a^j\right)}{\partial q_k}\ , \qquad j,\ k = 1, ..., n., \quad j \neq k, \tag{2.8}$$

which do not involve the unknown function $h(\mathbf{q})$. A system of partial differential equations must be solved, to find a function $\gamma$ (see examples 1 and 2 below).

If the integration factor $\gamma$ is zero then the constraint is considered to be not integrable and therefore the constraint is referred to as nonholonomic.

A classical example of a nonholonomic mechanical system, and very important in the study of wheeled mobile robots [SSVO09], is a disk rolling vertically without slipping on a horizontal plane.

**Example 1** *Consider a disk rolling vertically without slipping on a horizontal plane $\Sigma_1$, while keeping the plane that contains the disk $\Sigma_2$ in the vertical position (see Figure 2.2).*

*The posture coordinates, describing the motion of the disc with respect to the plane $\Sigma_1$ are the position given by $x$ and $y$, the Cartesian coordinates of the point $P$ (point of contact of the disc with plane $\Sigma_1$), and the angle $\theta$ characterizing the orientation of the disk with respect to the $x$ axis. So, the configuration vector is therefore $\mathbf{q} = (x, y, \theta)^T$. Once there is a restriction that the disk does not slip, the speed at point $P$ has zero component in the direction orthogonal to the plan $\Sigma_2$ .*

*The pure rolling constraint for the disk is expressed in the Pfaffian form as:*

$$\dot{x}\sin\theta - \dot{y}\cos\theta = [\sin\theta,\ -\cos\theta,\ 0]\,\dot{\mathbf{q}} = 0. \tag{2.9}$$

*Consider the pure rolling constraint* (2.9)*. The holonomy condition* (2.8) *gives:*

$$\begin{cases} \sin\theta\frac{\partial\gamma}{\partial y} = -\cos\theta\frac{\partial\gamma}{\partial x}, \\ \cos\theta\frac{\partial\gamma}{\partial\theta} = \gamma\sin\theta, \\ \sin\theta\frac{\partial\gamma}{\partial\theta} = -\gamma\cos\theta. \end{cases}$$

*Squaring and adding the last two equations gives*

$$\frac{\partial\gamma}{\partial\theta} = \pm\gamma.$$

*Assume, for example,* $\frac{\partial\gamma}{\partial\theta} = \gamma$ *(the same conclusion is reached by letting* $\frac{\partial\gamma}{\partial\theta} = -\gamma$*) and using the above equations leads to*

$$\gamma\cos\theta = \gamma\sin\theta,$$
$$\gamma\sin\theta = -\gamma\cos\theta,$$

*whose only solution is* $\gamma = 0$*. So, the constraint* (2.9) *is nonholonomic.*

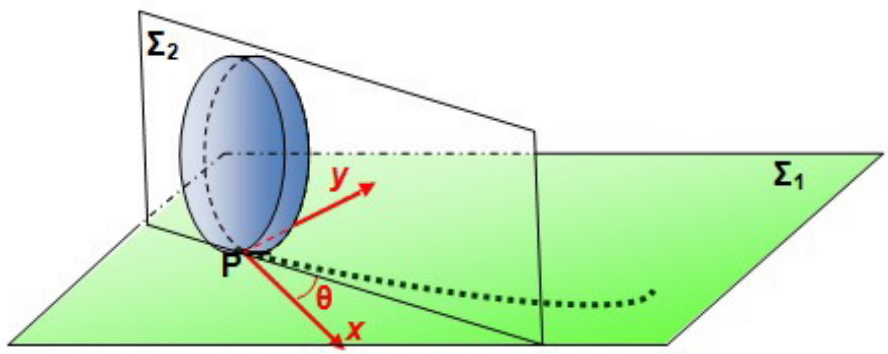


Figure 2.2: Rolling disk.

The situation is further complicated in the case of multiple Pfaffians constraints [MLS94]. When dealing with multiple kinematic constraints in the form (2.4), the nonholonomy of each constraint considered separately is not sufficient to infer that the whole set of constraints is nonholonomic. That is, given a set of $k$ constraints of the form (2.4), one needs to check whether each one of the $k$ constraints is integrable and, also, needs to check which independent linear combination of these constraints are integrable. In fact, it may happen that $p \leq k$ independent linear combinations

$$\sum_{i=1}^{k} \alpha_{ji}(\mathbf{q})\ a_i^T(\mathbf{q})\ \dot{\mathbf{q}}\ , \qquad j = 1, ..., p \leq k\ ,$$

are integrable. So, following [LO95]:

**Definition 2** *Given a set of* $k$ *constraints of the form* (2.4) *and let* $p$ *be the number of independent linear combination of these constraints that are integrable*

1. *If* $p = k$ *the system of kinematic constraints in the form* (2.4) *is completely integrable and the system is* ***holonomic*** *(see example 2).*

2. *If $p = 0$, i.e., there are no integrable independent linear combinations of the constraints, then the system is* ***completely nonholonomic****.*

3. *If $0 < p < k$ the system of kinematic constraints in the form* (2.4) *is nonholonomic, but the set* (2.4) *is integrable, so this situation is denoted as* ***partial nonholonomic****.*

**Example 2** *Consider the system of Pfaffian constraints*

$$\begin{cases} (\dot{x} + \dot{y})\ e^x + \dot{z} = 0, \\ \dot{y} + (\dot{x} + \dot{z})\ e^x = 0. \end{cases}$$

*Taken the first constraint, the holonomy condition (2.8) gives:*

$$\begin{cases} e^x \frac{\partial \gamma}{\partial y} = e^x \gamma + e^x \frac{\partial \gamma}{\partial x}, \\ e^x \frac{\partial \gamma}{\partial z} = \frac{\partial \gamma}{\partial x}, \\ e^x \frac{\partial \gamma}{\partial z} = \frac{\partial \gamma}{\partial y}. \end{cases} \tag{2.10}$$

*By substituing the second and the third equations into the first leads to* $e^x \gamma = 0$, *so,* $\gamma = 0$. *Therefore, the first constraint is nonholonomic. The same conclusion is reached if the second constraint is taken. So, taken separately, these two constraints are found to be non-integrable. However, subtracting the second from the first leads to:*

$$(e^x - 1)\,\dot{y} - (e^x - 1)\,\dot{z} = 0$$

*so, this leads to*

$$\dot{y} = \dot{z}.$$

*The system of constraints (2.10) is equivalent to:*

$$\begin{cases} \dot{y} = \dot{z}, \\ \dot{x} e^x + \dot{z}\ \ (e^x + 1) = 0. \end{cases}$$

*which can be integrated, giving*

$$\begin{cases} y - z = c_1, \\ \ln(e^x + 1) + z = c_2, \end{cases}$$

*where $c_1$ and $c_2$ are the integration constants. In this example, $p = k$, so the set of differential constraints is completely equivalent to a set of holonomic constraints then the system* (2.10) *is holonomic.*

As seen, there are difficulties in trying to decide about the holonomy or nonholonomy of a set of kinematic constraints. The integrability criteria can be obtained by a different viewpoint, i.e., not seeing the problem from the standpoint of the directions where movement is not possible, but rather the directions in where the movement is possible. So, it is convenient to convert problems with nonholonomic constraints into another form.

The set of $k$ Pfaffian constraints (2.4) defines, at each configuration $\mathbf{q}$, the admissible generalized velocities as those contained in the $(n-k)$-dimensional null space of matrix $A^T(\mathbf{q})$, $N\left(A^T(\mathbf{q})\right)$. Equivalently, if $\{g_1(\mathbf{q}), ..., g_{n-k}(\mathbf{q})\}$ is a basis for this space, all the feasible trajectories for the mechanical system are obtained as solutions of

$$\dot{\mathbf{q}} = \mathbf{g}_1(\mathbf{q})u_1 + ... + \mathbf{g}_m(\mathbf{q})u_m = G(\mathbf{q})\,\mathbf{u}, \qquad m = n-k, \tag{2.11}$$

for arbitrary $\mathbf{u}(t)$.

This may be regarded as a *nonlinear control system* with the state vector $\mathbf{q} = (q_1, q_2, ..., q_n)^T \in Q \subset \mathbb{R}^n$, the vector of the control variables (the inputs) $\mathbf{u} = (u_1, u_2, ..., u_m)^T \in U \subset \mathbb{R}^m$ and $\mathbf{g}_i,\ i = 1, ..., m$, are specified vector fields [3].

A *feasible trajectory* $\mathbf{q}(t)$ for the system must satisfy the equation (2.11) for some value of the controls $\mathbf{u}(t) \in \mathbb{R}^m$. In particular, the system (2.11) is said to be *driftless* because if the inputs are zero then $\dot{\mathbf{q}} = 0$. It is refered to as the *kinematic model* of the constrained mechanical system, because it is possible to choose the basis of $N\left(A^T(\mathbf{q})\right)$ in such a way that the inputs $\mathbf{u} = (u_1, u_2, ..., u_m)^T$ have a physical interpretation.

The holonomy or nonholonomy of the constraints (2.4) can be established by analyzing the controllability properties (see next section) of the associated kinematic model (2.11). In general terms, controllability is the ability to steer a system from a given initial state to any final state, in finite time, using the available controls. So, using [SSVO09]:

1. If the system (2.11) is **controllable**, given two arbitrary configurations $\mathbf{q}^i$ and $\mathbf{q}^f$ in $Q$, there exists a choice of $\mathbf{u}$ that steers the system from $\mathbf{q}^i$ to $\mathbf{q}^f$. That is, there exists a trajectory that joins the two configurations and satisfies the kinematic constraints (2.4). Therefore, these do not affect in any way the accessibility of $Q$, and they are (completely) nonholonomic.

2. If the system (2.11) is **not controllable**, the kinematic constraints (2.4) reduce the set of accessible configurations in $Q$. Hence, the constraints are partially or completely integrable depending on the dimension of the accessible configuration space. Suppose that $d\ (< n)$ is the dimension of the accessible configuration space. In particular:

   - If $m < d < n$, the loss of accessibility is not maximal, and thus constraints (2.4) are partially integrable. The mechanical system is still nonholonomic.
   - If $d = m$, the loss of accessibility is maximal, and the constraints (2.4) are completely integrable. Therefore, the mechanical system is holonomic.

[3] A (smooth) *vector field* $\mathbf{f}$ on $Q$ is a (smooth) mapping that assigns to each point a vector tangent to $Q$ at that point:

$$\mathbf{q} \in Q \longmapsto \mathbf{f}(\mathbf{q}) \in T_{\mathbf{q}}Q.$$

In this context, we can study the nature of Pfaffian constraint (2.6) by studying the controllability properties of equation (2.11). That is, the constraint is completely nonholonomic if the corresponding control system can be steered between any two points. Thus, the reachable configurations of the system are not restricted. Conversely, if the constraint is holonomic, then all motions of the system must lie on an appropriate constraint surface and the corresponding control system can only be steered between points on the given manifold.

The equivalence between controllability and nonholonomy can be show by exhibiting a reconfiguration maneuver in the example of the rolling disk, (see Figure 2.3).

**Example 3** *Through the following sequence of movements, that do not violate the constraint* (2.9)*, the disk can be driven from any initial configuration* $\mathbf{q}^i = (x_i, y_i, \theta_i)^T$ *to any final configuration* $\mathbf{q}^f = (x_f, y_f, \theta_f)^T$.

1. *Rotate the disk around its vertical axis so as to reach the orientation* $\theta$*, for which the initial contact point* $(x_i, y_i)$ *goes through the final contact point* $(x_f, y_f)$.

2. *Roll the disk on the plane at a constant orientation* $\theta$*, until the contact point reaches its final position* $(x_f, y_f)$.

3. *Rotate again the disk around its vertical axis to change the orientation from* $\theta$ *to* $\theta_f$.

*In conclusion, the constraint* (2.9) *implies no loss of accessibility in the configuration space of the disk (i.e., the disk is controllable) and it is nonholonomic.*

To effectively make use of this approach, it is necessary to have practical controllability conditions to verify for the nonlinear control system (2.11).

For this purpose, we shall present, in the next section, tools from control theory based on differential geometry. The properties of a control system depend on the properties of vector fields and interactions between them. The basic tool to understand such interactions is their Lie bracket and they are applicable to one important class of nonlinear control systems, the *affine control systems:*

**Definition 3** *Consider the system on the form:*

$$\dot{\mathbf{q}} = \mathbf{h}(\mathbf{q}) + \sum_{j=1}^{m} \mathbf{g}_j(\mathbf{q}) u_j, \tag{2.12}$$

*or in a compact way:*

$$\dot{\mathbf{q}} = \mathbf{f}(\mathbf{q}, \mathbf{u}), \tag{2.13}$$

*where* $\mathbf{q} = (q_1, q_2, ..., q_n)^T \in \mathbb{R}^n$ *are local coordinates for a smooth manifold* $Q$ *(the state space manifold),* $\mathbf{u} = (u_1, u_2, ..., u_m)^T \in U \subset \mathbb{R}^m$ *are the control variables, and the mappings*

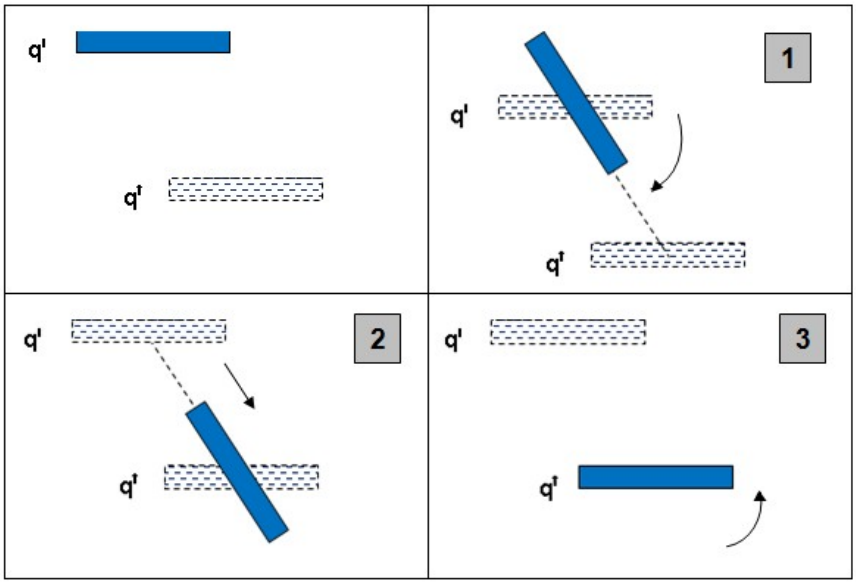


Figure 2.3: Disk reconfiguration manoeuvre.

$\mathbf{h}, \mathbf{g}_1, \mathbf{g}_2, ..., \mathbf{g}_m$ *are smooth vector fields on* $Q$*. These systems are called* ***affine control systems***[4]*, the vector field* $\mathbf{h}$ *is called the* ***drift vector field*** *and* $\mathbf{g}_j$ $(j = 1, ..., m)$ *are referred to as the* ***control vector fields****.*

### 2.2.1 Controllability

Roughly speaking, controllability is the ability to steer a system from a given initial state to any final state, in finite time, using the available controls. Some key results are provided below to be able to answer very general questions about the possibility of steering $\mathbf{q}^i$ to $\mathbf{q}^f$ (given arbitrary points) by using admissible controls over a finite time interval.

**Definition 4** *[NvdS90] The nonlinear system (2.12) is said to be* ***controllable*** *if, for any two points* $\mathbf{q}^i, \mathbf{q}^f$ *in* $Q$*, there exist a finite time* $T \geq t_0 \geq 0$ *and an admissible control function* $u : [t_0, T] \to U$ *such that the unique solution of (2.12) with initial condition* $q(t_0) = \mathbf{q}^i$ *at time* $t = T$ *and with input function* $u(\cdot)$ *satisfies* $q(T) = \mathbf{q}^f$*.*

[4]Linear systems $\dot{\mathbf{q}} = A\mathbf{q} + B\mathbf{u}$, with, $\mathbf{q} \in \mathbb{R}^n$, $\mathbf{u} \in \mathbb{R}^m$, $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{m\times n}$ are a particular case of affine control systems, since they may be writen as

$$\dot{\mathbf{q}} = \underbrace{A\mathbf{q}}_{\mathbf{h}(\mathbf{q})} + \sum_{j=1}^{m} u_j \underbrace{Be_j}_{g_j(\mathbf{q})}.$$

Let $\mathbf{q}^i$ be an arbitrary point in $Q$, $V$ a given neighborhood of $\mathbf{q}^i$ in $Q$ and $\tau > 0$.

**Definition 5** *$R^V(\mathbf{q}^i, \tau)$ is the **set of states reachable at time** $\tau$ **from** $\mathbf{q}^i$ following admissible trajectories contained in $V$ for $t \leq \tau$.*

That is,

$$\begin{aligned} R^V(\mathbf{q}^i, \tau) \;=\; & \{\mathbf{q} \in Q : \text{ there exists an admissible input } u : [t_0, \tau] \to U \\ & \text{such that the evolution of (2.12) for } \mathbf{q}(t_0) = \mathbf{q}^i \text{ satisfies} \\ & \mathbf{q}(t) \in V,\ t_0 \leq t \leq \tau, \text{ and } \mathbf{q}(\tau) = \mathbf{q}\}. \end{aligned}$$

This definition leads to the concept of reachable set from $\mathbf{q}^i$ during the interval $[0, T]$ as follows:

**Definition 6** *$R_T^V(\mathbf{q}^i)$ is the set of **states reachable within time** $T$ **from** $\mathbf{q}^i$ with trajectories contained in the neighbourhood $V$.*

That is,

$$R_T^V(\mathbf{q}^i) = \bigcup_{0 < \tau \leq T} R^V(\mathbf{q}^i, \tau).$$

**Definition 7** *[NvdS90] The system (2.12) is said to be **locally accessible** from $\mathbf{q}^i$ if $R_T^V(\mathbf{q}^i)$ contains a non-empty open subset of $Q$ for all neighborhoods $V$ of $\mathbf{q}^i$ and all $T > 0$.*

By performing a simple algebraic test, local accessibility is easily checked. For this purpose, the definition of the accessibility algebra $C$ is needed. But first let us introduce the Lie algebra and Lie brackets definitions:

**Definition 8** *A **Lie algebra** is a vector space $V_L$ together with a map*

$$[.\ ,.] : V_L \times V_L \to V_L$$

*called the **Lie bracket**, such that it satisfies the following properties: consider $a, b, c \in V_L$*

1. *Bilinearity*

$$\begin{aligned} [a + \lambda b, c] &= [a, c] + \lambda [b, c], \\ [a, b + \lambda c] &= [a, b] + \lambda [a, c]. \end{aligned}$$

2. *Skew-symmetry:*

$$[a, b] = -[b, a].$$

*3. Jacobi identity*

$$[a,[b,c]]+[c,[a,b]]+[b,[c,a]]=0.$$

*The Lie algebra of the vector fields on $Q$ is denoted by $Lie\,(Q)$. A subspace of $Lie(Q)$ is a* ***Lie subalgebra*** *if it is closed under the Lie bracket.*

The Lie bracket for two vector fields $\mathbf{f}$ and $\mathbf{g}$ in $\mathbb{R}^n$ can be computed through the following formula:

$$[\mathbf{f},\mathbf{g}]\,(\mathbf{q})=\frac{\partial\mathbf{g}}{\partial\mathbf{q}}\mathbf{f}\,(\mathbf{q})-\frac{\partial\mathbf{f}}{\partial\mathbf{q}}\mathbf{g}\,(\mathbf{q})$$

where $\frac{\partial\mathbf{g}}{\partial\mathbf{q}}$ and $\frac{\partial\mathbf{f}}{\partial\mathbf{q}}$ are the Jacobian matrices of $g$ and $f$ respectively.

**Definition 9** *The* ***accessibility algebra C*** *for (2.12) is the smallest subalgebra of $Lie(Q)$ that contains $\mathbf{h},\mathbf{g}_1,\mathbf{g}_2,...,\mathbf{g}_m$.*

By definition, all the Lie brackets that can be generated using the vector fields $\mathbf{h},\mathbf{g}_1,\mathbf{g}_2,...,\mathbf{g}_m$, belong to $C$.

A distribution[5] is the subspace generated by a collection of vector fields.

**Definition 10** *[SSVO09] The* ***accessibility distribution*** $\Delta_C$ *is defined as*

$$\Delta_C(\mathbf{q})=\{v(\mathbf{q}):\mathbf{v}\in\mathbf{C}\}\,,\quad \mathbf{q}\in Q.$$

*In other words, $\Delta_C$ is the involutive closure of $\Delta=span\,\{\mathbf{h},\mathbf{g}_1,\mathbf{g}_2,...,\mathbf{g}_m\}$.*

For a distribution of finite dimensions it is sufficient to check if the Lie brackets of the basic-elements are contained in the distribution.

The computation of $\Delta_C$ may be organized as an iterative procedure [SSVO09]:

$$\Delta_C=span\,\{\mathbf{v}:\mathbf{v}\in\Delta_i,\quad\forall i\geq 1\}\,,$$

[5] A **distribution** $\Delta$ associated with the $k$ vector fields $\{\boldsymbol{\xi}_1,\boldsymbol{\xi}_2,...,\boldsymbol{\xi}_k\}$ is the mapping that assigns to each point $\mathbf{q}\in\mathbb{R}^n$ the subspace of the tangent space $T_{\mathbf{q}}\,(\mathbb{R}^n)$ defined as $\Delta(\mathbf{q})=span\,\{\boldsymbol{\xi}_1(\mathbf{q}),\boldsymbol{\xi}_2(\mathbf{q}),...,\boldsymbol{\xi}_m(\mathbf{q})\}$ or, in a short way $\Delta=span\,\{\boldsymbol{\xi}_1,\boldsymbol{\xi}_2,...,\boldsymbol{\xi}_m\}$.

A distribution is said to be **regular** if the dimension of the subspace does note vary with $\mathbf{q}$, i.e., $dim\,\Delta(\mathbf{q})=d$, with $d$ constant for all $\mathbf{q}$. In this case, d is called the **dimension** *of the distribution.*

A distribution is **involutive** if it is closed under the Lie bracket, i.e.,

$$\Delta \text{ is involutive} \quad\Leftrightarrow\quad \forall\boldsymbol{\xi}_1,\boldsymbol{\xi}_2\in\Delta,\quad[\boldsymbol{\xi}_1,\boldsymbol{\xi}_2]\in\Delta.$$

The **involutive closure** of a distribution, denoted by $\bar{\Delta}$, is the smallest distribution containing $\Delta$ such that $\mathbf{f},\mathbf{g}\subset\bar{\Delta}$ then $[\mathbf{f},\mathbf{g}]\subset\bar{\Delta}$.

with

$$\begin{aligned}\Delta_1 &= \Delta = span\left\{\mathbf{h}, \mathbf{g}_1, \mathbf{g}_2, ..., \mathbf{g}_m\right\}, \\ \Delta_i &= \Delta_{i-1} + span\left\{[\boldsymbol{\rho}, \boldsymbol{\varphi}] : \boldsymbol{\rho} \in \Delta_1, \quad \boldsymbol{\varphi} \in \Delta_{i-1}\right\}, \quad i \geq 2.\end{aligned}$$

This procedure stops after $\kappa$ steps, where $\kappa$ is the smallest integer such that

$$\Delta_{\kappa+1} = \Delta_\kappa = \Delta_C.$$

This number is called *nonholonomy degree* of the system and is related to the "level of Lie brackets that must be included in $\Delta_C$.

Consider a system without drift, i.e., in (2.12) $\mathbf{h} = 0$,

$$\dot{\mathbf{q}} = \sum_{j=1}^{m} \mathbf{g}_j(\mathbf{q}) u_j. \tag{2.14}$$

The system is controllable if and only if the condition (called the *accessibility rank condition*)

$$\dim \Delta_C(\mathbf{q}) = n, \tag{2.15}$$

holds[6].

The following cases may occur [SSVO09]:

1. If (2.15) does not hold, the system (2.11) is not controllable and the kinematic constraints (2.4) are at least partially integrable. In particular:

   (a) if
   $$m < \dim \Delta_C(\mathbf{q}) < n,$$
   the constraints (2.4) are partially integrable (the mechanical system is still nonholonomic);

   (b) if
   $$\dim \Delta_C(\mathbf{q}) = m,$$
   they are completely integrable and, hence holonomic. This happens when $\Delta_C$ coincides with $\Delta = span\left\{\mathbf{g}_1, ...\mathbf{g}_m\right\}$, i.e., when $\Delta_C$ is involutive (*Frobenius theorem* [7]). Therefore, the mechanical system is holonomic.

2. If (2.15) holds the system (2.11) is controllable and the kinematic constraints (2.4) are (completely) nonholonomic.

[6] For a linear systems $\dot{\mathbf{q}} = A\mathbf{q} + B\mathbf{u}$, (2.15) becomes:

$$rank\left(\left[\mathbf{B} \;\; \mathbf{AB} \;\; \mathbf{A}^2 B \;\; ... \;\; \mathbf{A}^{n-1} B\right]\right) = n,$$

i.e., the necessary and sufficient condition for controllability due to Kalman.

[7]

**Theorem 2** *[MLS94] (**Frobenius' theorem**). A regular distribution is integrable if and only if it is involutive.*

**Application to a differential drive mobile robot**

Consider the differential-drive mobile robot car in Figure 2.4.

The Cartesian coordinates $(x, y)$ are the position in the plane of the midpoint of the axle connecting the rear wheels, $L$ is the axis lenght of the rear wheels and $\theta$ denotes the heading angle measured anticlockwise from the $x$-axis. The controls $u_1$ and $u_2$ are the angular velocity of the right wheel and of the left wheel respectively, with $u_1(t), u_2(t) \in [-u_{\max}, u_{\max}]$. If the same velocity is applied to both wheels, the robot moves along a straight line (maximum forward velocity if $u_1 = u_2 = u_{\max}$). The robot can turn by choosing $u_1 \neq u_2$ (if $u_1 = -u_2 = u_{\max}$ the robot turns anticlockwise around the midpoint of the axle).

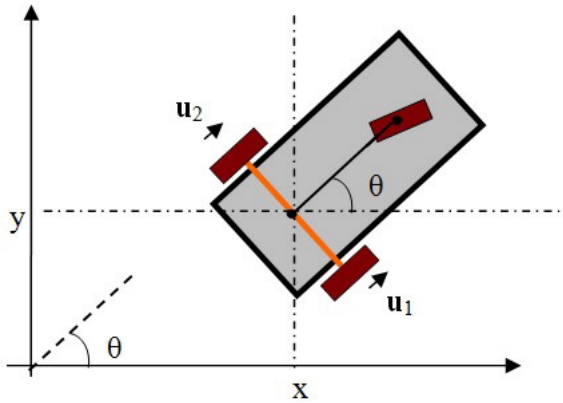


Figure 2.4: Differential-drive mobile robot car.

The velocity control of the two rear wheels leads to the velocity of translation of the robot:

$$v = \frac{u_1 + u_2}{2},$$

and the angular velocity:

$$w = \frac{u_1 - u_2}{L},$$

where $L$ is the distance between the rear wheels. So, the forward/backward motion and the clockwise/counterclockwise rotation are the two kinds of motion.

With this model, the rolling without slipping constraint is expressed by the following equation:

$$\dot{x} \sin\theta - \dot{y} \cos\theta = 0. \tag{2.16}$$

Writting this equation in the form of Pfaffin constraints, with $\mathbf{q} = (x, y, \theta)$ we have:

$$\mathbf{a}^T(\mathbf{q})\,\dot{\mathbf{q}} = [-\sin\theta,\ \cos\theta,\ 0]\,\dot{\mathbf{q}} = \mathbf{0}.$$

Consider the matrix

$$\mathbf{G}(\mathbf{q}) = [\mathbf{g}_1(\mathbf{q}), \mathbf{g}_2(\mathbf{q})] = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix},$$

whose columns $\mathbf{g}_1(\mathbf{q}) = \begin{bmatrix} \cos\theta & \sin\theta & 0 \end{bmatrix}^T$ and $\mathbf{g}_2(\mathbf{q}) = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$ are, for each $\mathbf{q}$, a basis of the null space of the matrix associated with the Pfaffian constraint, $N\left(A^T(\mathbf{q})\right)$. All the admissible generalized velocities at $\mathbf{q}$ are therefore obtained as a linear combination of $\mathbf{g}_1(\mathbf{q})$ and $\mathbf{g}_2(\mathbf{q})$.

These motions combine in an admissible way and give the following driftless affine control system:

$$\dot{\mathbf{q}} = \underbrace{v\ \mathbf{g}_1(\mathbf{q}) + w\ \mathbf{g}_2(\mathbf{q})}_{\mathbf{f}(\mathbf{q},\mathbf{u})},$$

(the choice of a basis of the null space of the matrix associated with the Pfaffian constraint is not unique, in this case we have chosen the velocity of translation and the angular velocity of our robot).

The vector field $\mathbf{g}_1$ generates the forward/backward motion, the vector field $\mathbf{g}_2$ generates the clockwise/counterclockwise rotation, and the vector field $[\mathbf{g}_1, \mathbf{g}_2]$ generates the motion in the direction perpendicular to the orientation of the car.

Supose $L = 1$ without lost of generality. So, the differential-drive mobile robot car in Figure 2.4 can be represented by the model:

$$\begin{cases} \dot{x} &= v\cos\theta, \\ \dot{y} &= v\sin\theta, \\ \dot{\theta} &= w, \end{cases} \tag{2.17}$$

or in a compact form:

$$\dot{\mathbf{q}}(t) = \mathbf{f}(\mathbf{q}(t), \mathbf{u}(t)), \tag{2.18}$$

where $\mathbf{q} = [x\ \ y\ \ \theta]^T$ e $\mathbf{u} = [v\ \ w]^T$.

The forbidden direction of this system corresponds to the direction of the Lie bracket of the control vector fields, as can easily be seen by direct computation.

$$\begin{aligned} [\mathbf{g}_1, \mathbf{g}_2] &= \tfrac{\partial g_2}{\partial q} g_1 - \tfrac{\partial g_1}{\partial q} g_2 \\ &= \begin{bmatrix} \frac{\partial g_2}{\partial x} & \frac{\partial g_2}{\partial y} & \frac{\partial g_2}{\partial \theta} \end{bmatrix} g_1 - \begin{bmatrix} \frac{\partial g_1}{\partial x} & \frac{\partial g_1}{\partial y} & \frac{\partial g_1}{\partial \theta} \end{bmatrix} g_2 \\ &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \cos\theta \\ \sin\theta \\ 0 \end{bmatrix} - \begin{bmatrix} 0 & 0 & -\sin\theta \\ 0 & 0 & \cos\theta \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} -\sin\theta \\ \cos\theta \\ 0 \end{bmatrix}. \end{aligned}$$

The Lie bracket of the two input vector field does not belong to $span\{\mathbf{g}_1, \mathbf{g}_2\}$, as a consequence, the accessibility distribution $\Delta_C$ has dimension 3.

$$\dim \Delta_C = \dim span\left\{\mathbf{g}_1, \mathbf{g}_2, [\mathbf{g}_1, \mathbf{g}_2]\right\} = 3.$$

So, the nonlinear system (2.17) is controllable, and the constraint (2.16) is nonholonomic.

**Conclusion**: this system is subject to a nonholonomic constraint (2.16) since it cannot move in the direction perpendicular to the orientation of the car. However, the system is controllable, which means that the forbidden motion must be able to be generated from the allowable ones.

When trying to obtain a linearization [8] of this system around any operating point with zero velocity

$$\mathbf{q} = (x_0;\ y_0; \theta_0) \ \text{ with } \mathbf{u} = (0, w_0)\,,$$

the resulting linear system :

$$\left\{\begin{array}{lcl} \dot{x} & = & 0 \\ \dot{y} & = & 0 \\ \dot{\theta} & = & w \end{array}\right. \quad \text{, that is,} \quad \dot{\mathbf{q}} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix}$$

is not controllable

$$rank\left(\left[\mathbf{B} \ \ \mathbf{AB} \ \ \mathbf{A}^2 B\right]\right) = rank\left(\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}\right) = 1 \neq 3.$$

Therefore, linear control methods, or any auxiliary procedure based on linearization, cannot be used to stabilize this system.

### Application to a car-like robot

Consider the car-like vehicle in Figure 2.5. This vehicle has two fixed wheels mounted on a rear axle and two steerable wheels mounted on a front axle. In this example, the heading is controlled by the angle of the two front wheels. The Cartesian coordinates $(x, y)$ are the position in the plane of the midpoint of the axle connecting the rear wheels, $L$ is the distance between the rear and front wheels axles, $\theta$ is the orientation of the vehicle, and $\phi$ denotes the steering angle.

The forward/backward motion is achieved by applying a velocity to the rear-wheels, a car-like robot moves along its longitudinal axis, or rather, it rotates about the instantaneous rotation centre determined by the steering angle. The clockwise/counterclockwise rotation is achieved

[8] For many nonlinear systems, linear approximations can be used for a first analysis in the synthesis of control law. The linearization can also provide some indication of the controllability and stability of nonlinear system. More precisely, if the linearized system is controllable and stable, then the original nonlinear system is controllable and stable, at least locally. However, the reverse cannot be applied.

by turning the front-wheels, the direction of the motion generated by the driving wheels of a car-like robot can be changed.

The rotation is determined by the angle of the front wheels with the respect to heading ($\phi$) and distance between the instantaneous centre of rotation (midpoint of the rear axle) and the midpoint of the front axle ($L$). Some simple trigonometry, shows that the *instantaneous rotation radius* $R$ is given by $R = L/\tan(\phi)$ (see Figure 2.6).

The car-like robot in Figure 2.5 can be represented by the model:

$$\begin{cases} \dot{x} &= v\cos\theta, \\ \dot{y} &= v\sin\theta, \\ \dot{\theta} &= v \cdot c, \end{cases} \tag{2.19}$$

with control inputs $v$ and $c$ satisfying $v \in [-v_{\max}, v_{\max}]$ and $c \in [-c_{\max}, c_{\max}]$. Note that the modulus of $c$ is the inverse of the turning radius and the minimum turning radius $R_{\min} = |c_{\max}|^{-1}$.

In contrast with the previous example, this vehicle cannot turn with zero velocity and, in addition, it has a minimum turning radius. A car-like mobile robot must drive forward or backwards if it wants to turn but a differentially-driven robot can turn on the spot by giving opposite speeds to both wheels.

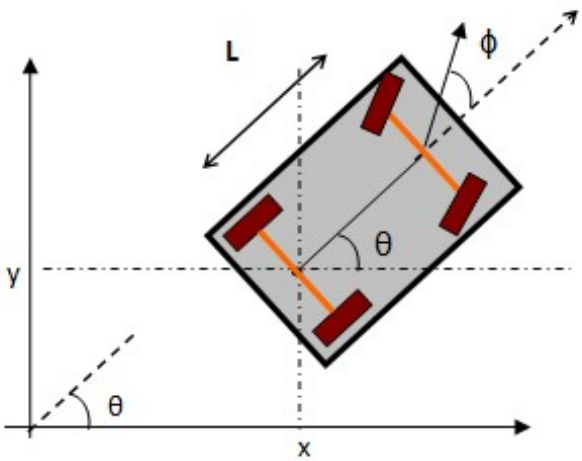


Figure 2.5: Car-like.

With this modeling, the rolling without slipping constraints of the front and rear wheels are expressed by the following equations:

$$\begin{aligned} &\dot{x}\sin\theta - \dot{y}\cos\theta = 0, \\ &\dot{x}\sin(\theta+\phi) - \dot{y}\cos(\theta+\phi) - L\dot{\theta}\cos\phi = 0. \end{aligned} \tag{2.20}$$

The constraints for the front and rear wheels are formed by setting the sideways velocity of the wheels to zero. The velocity of the back wheels perpendicular to their direction is $\dot{x}\sin\theta - \dot{y}\cos\theta$ and the velocity of the front wheels perpendicular to the direction they are pointing to is $\dot{x}\sin(\theta+\phi) - \dot{y}\cos(\theta+\phi) - L\dot{\theta}\cos\phi$.

Writting this equation in the form of Pfaffin constraints, with $\mathbf{q} = (x, y, \theta)$ we have:

$$\mathbf{A}^T(\mathbf{q})\,\dot{\mathbf{q}} = \begin{bmatrix} \sin\theta & \cos\theta & 0 & 0 \\ \sin(\theta+\phi) & \cos(\theta+\phi) & L\cos\phi & 0 \end{bmatrix} \dot{\mathbf{q}} = \mathbf{0}.$$

Consider the matrix

$$\mathbf{G}(\mathbf{q}) = [\mathbf{g}_1(\mathbf{q}), \mathbf{g}_2(\mathbf{q})] = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ \frac{1}{L}\tan\phi & 0 \\ 0 & 1 \end{bmatrix}, \quad \phi \neq \pm\frac{\pi}{2},$$

whose columns $\mathbf{g}_1(\mathbf{q}) = \begin{bmatrix} \cos\theta & \sin\theta & \frac{1}{L}\tan\phi & 0 \end{bmatrix}^T$ and $\mathbf{g}_2(\mathbf{q}) = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T$ are, for each $\mathbf{q}$, a basis of the null space of the matrix associated with the Pfaffian constraint, $N\left(A^T(\mathbf{q})\right)$. All the admissible generalized velocities at $\mathbf{q}$ are therefore obtained as a linear combination of $\mathbf{g}_1(\mathbf{q})$ and $\mathbf{g}_2(\mathbf{q})$.

Supose $L = 1$ without lost of generality. These motions combine in an admissible way and give the following driftless affine control system $\mathbf{q} = (x, y, \theta, \phi)$:

$$\dot{\mathbf{q}} = \underbrace{v\ \mathbf{g}_1(\mathbf{q}) + w\ \mathbf{g}_2(\mathbf{q})}_{\mathbf{f}(\mathbf{q},\mathbf{u})},$$

where, $v$ and $w$ are, respectively, the driving and the steering velocity, and

$$\mathbf{g}_1(\mathbf{q}) = \begin{bmatrix} \cos\theta \\ \sin\theta \\ \tan\phi \\ 0 \end{bmatrix} \quad \text{and} \quad \mathbf{g}_2(\mathbf{q}) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}.$$

So, the car-like vehicle in Figure 2.5 can be represented by the model ($\phi \neq \pm\frac{\pi}{2}$):

$$\begin{cases} \dot{x} &= v\cos\theta, \\ \dot{y} &= v\sin\theta, \\ \dot{\theta} &= v\tan\phi, \\ \dot{\phi} &= w. \end{cases} \tag{2.21}$$

To test controllability we compute the first two Lie brackets $[\mathbf{g}_1, \mathbf{g}_2]$ and $[\mathbf{g}_1, [\mathbf{g}_1, \mathbf{g}_2]]$:

$$\begin{aligned}
[\mathbf{g}_1, \mathbf{g}_2] &= \tfrac{\partial g_2}{\partial q} g_1 - \tfrac{\partial g_1}{\partial q} g_2 \\
&= \begin{bmatrix} \frac{\partial g_2}{\partial x} & \frac{\partial g_2}{\partial y} & \frac{\partial g_2}{\partial \theta} & \frac{\partial g_2}{\partial \phi} \end{bmatrix} g_1 - \begin{bmatrix} \frac{\partial g_1}{\partial x} & \frac{\partial g_1}{\partial y} & \frac{\partial g_1}{\partial \theta} & \frac{\partial g_1}{\partial \phi} \end{bmatrix} g_2 \\
&= \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \cos\theta \\ \sin\theta \\ \tan\phi \\ 0 \end{bmatrix} - \begin{bmatrix} 0 & 0 & -\sin\theta & 0 \\ 0 & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & \sec^2\phi \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \\
&= \begin{bmatrix} 0 \\ 0 \\ -\sec^2\phi \\ 0 \end{bmatrix},
\end{aligned}$$

$$\begin{aligned}
[\mathbf{g}_1, [\mathbf{g}_1, \mathbf{g}_2]] &= \tfrac{\partial g_2}{\partial q} g_1 - \tfrac{\partial g_1}{\partial q} [\mathbf{g}_1, \mathbf{g}_2] \\
&= \begin{bmatrix} \frac{\partial [\mathbf{g}_1, \mathbf{g}_2]}{\partial x} & \frac{\partial [\mathbf{g}_1, \mathbf{g}_2]}{\partial y} & \frac{\partial [\mathbf{g}_1, \mathbf{g}_2]}{\partial \theta} & \frac{\partial [\mathbf{g}_1, \mathbf{g}_2]}{\partial \phi} \end{bmatrix} g_1 - \begin{bmatrix} \frac{\partial g_1}{\partial x} & \frac{\partial g_1}{\partial y} & \frac{\partial g_1}{\partial \theta} & \frac{\partial g_1}{\partial \phi} \end{bmatrix} [\mathbf{g}_1, \mathbf{g}_2] \\
&= \begin{bmatrix} -\sin\theta \sec^2\phi \\ \cos\theta \sec^2\phi \\ 0 \\ 0 \end{bmatrix},
\end{aligned}$$

where,

$$\begin{bmatrix} \dfrac{\partial [\mathbf{g}_1, \mathbf{g}_2]}{\partial x} & \dfrac{\partial [\mathbf{g}_1, \mathbf{g}_2]}{\partial y} & \dfrac{\partial [\mathbf{g}_1, \mathbf{g}_2]}{\partial \theta} & \dfrac{\partial [\mathbf{g}_1, \mathbf{g}_2]}{\partial \phi} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\sec^2\phi \tan\phi \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

and

$$\begin{bmatrix} \dfrac{\partial g_1}{\partial x} & \dfrac{\partial g_1}{\partial y} & \dfrac{\partial g_1}{\partial \theta} & \dfrac{\partial g_1}{\partial \phi} \end{bmatrix} = \begin{bmatrix} 0 & 0 & -\sin\theta & 0 \\ 0 & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & \sec^2\phi \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

The Lie brackets $[\mathbf{g}_1, \mathbf{g}_2]$ and $[\mathbf{g}_1, [\mathbf{g}_1, \mathbf{g}_2]]$ do not belong to $span\{\mathbf{g}_1, \mathbf{g}_2\}$ and are linearly independent , as a consequence, the accessibility distribution $\Delta_C$ has dimension 4.

$$\dim \Delta_C = \dim span \left\{ \mathbf{g}_1, \mathbf{g}_2, [\mathbf{g}_1, \mathbf{g}_2], [\mathbf{g}_1, [\mathbf{g}_1, \mathbf{g}_2]] \right\} = 4.$$

In conclusion, the nonlinear system (2.21) is controllable. and the constraints 2.20 are nonholonomic.

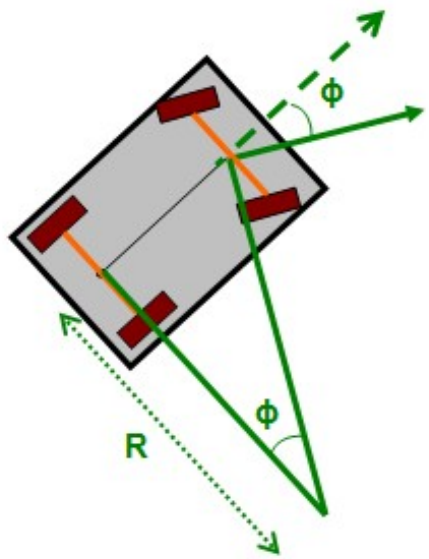


Figure 2.6: Instantaneous rotation radius.

### 2.2.2 Stability

#### I - Stability of dynamical systems

There exist Lyapunov stability theorems for proving the uniform asymptotic stability of autonomous and nonautonomous dynamical systems. For autonomous systems, the asymptotic stability of an equilibrium point can be shown by constructing a Lyapunov function $V$ in the neighborhood of the equilibrium point whose derivative $\dot{V}$ is negative definite.

For nonautonomous systems, the uniform asymptotic stability is guaranteed by imposing an additional constraint - the Lyapunov function has to be decreasing. In reality, we very often come across Lyapunov functions whose derivatives are only negative semi-definite. For autonomous systems it may then still be possible to conclude the asymptotic stability using LaSalle's invariant set theorem, provided that we can show the maximum invariant set to contain only the equilibrium point. It is easy and always possible to identify the set of points where the derivative of the Lyapunov function vanishes (i.e., converges to zero), but the maximum invariant set is only a subset of this set. The main challenge of LaSalle's theorem is therefore to sort out the maximum invariant set. This task may not be simple for complex systems.

For the general class of nonautonomous systems, invariant set theorems do not exist. This is because the positive limit set of the trajectories of nonautonomous systems are not invariant sets in general. Naturally, the asymptotic stability analysis of nonautonomous systems is more difficult. This difficulty is often overcome by the use of Barbalat's lemma. When applied to a Lyapunov function $V$ whose derivative is negative semi-definite and uniformly continuous,

Barbalat's lemma predicts the convergence of $\dot{V}$ to zero, asymptotically. Though this lemma does not establish the convergence of $V$ to zero, it often leads to satisfactory solution of control problems.

### Concepts of stability

As a premise to the study of the stabilization problem for the control system (2.12) the fundamental definitions are given and the concept of Lyapunov stability is introduced. Most of the definitions can be found in [SL91], but they are adapted to the concepts of stability from [CLSS97].

An equilibrium point is locally *Lyapunov stable* if all solutions which start near $\mathbf{q}^*$ (i.e., the initial conditions are in a neighborhood of $\mathbf{q}^*$) remain near $\mathbf{q}^*$ for all time. The equilibrium point $\mathbf{q}^*$ is said to be locally asymptotically stable if $\mathbf{q}^*$ is Lyapunov stable and, furthermore, all solutions starting near $q^*$ tend towards $q^*$ as $t \to \infty$.

By shifting the origin of the system, we may assume that the equilibrium point of interest occurs at $\mathbf{q}^* = \mathbf{0}$.

Consider the system of the form:

$$\dot{\mathbf{q}} = \mathbf{f}\left(\mathbf{q},t\right). \tag{2.22}$$

The nonlinear system (2.22) is said to be **autonomous** if $\mathbf{f}$ does not depend explicity on time, i.e., if the system state equation can be written as

$$\dot{\mathbf{q}} = \mathbf{f}\left(\mathbf{q}\right); \tag{2.23}$$

otherwise, the system is called **nonautonomous** [SL91][9].

Next, we provide some background definitions concerning nonlinear systems and equilibrium points, we define these stability concepts formally, for autonomous and nonautonomous systems. The concepts of stability for nonautonomous systems are quite similar to those of autonomous systems. However, due to the dependence of nonautonomous system behavior on initial time $t_0$, the definitions of these stability concepts include $t_0$ explicitly.

Lyapunov stability (also called stability) means that the system trajectory can be kept arbitrarily close to the origin by starting sufficiently close to it. More formally:

**Definition 11** *[SL91]*

1. *The equilibrium point* $\mathbf{0}$ *of the autonomous system (2.23) (i.e.,* $\mathbf{f}\left(\mathbf{0}\right) = \mathbf{0}$*) is* ***Lyapunov stable*** *if for any* $R > 0$*, there exists* $r > 0$*, such that*

$$\|\mathbf{q}\left(0\right)\| < r \quad \Rightarrow \quad \|\mathbf{q}\left(t\right)\| < R \qquad \forall t \geq 0. \tag{2.24}$$

[9] It should be noted that the term autonomous is also used in the robotics literature, with a different meaning, to categorize systems that can take decisions without human intervention.

2. *The equilibrium point* $\mathbf{0}$ *of the nonautonomous system (2.22) (i.e.,* $\mathbf{f}(\mathbf{0}, t) = \mathbf{0}$ *,* $\forall t \geq t_0$*,) is* ***Lyapunov stable*** *at time* $t_0$ *if for any* $R > 0$*, there exists a positive scalar function* $r = r(R, t_0)$ *, such that*

$$\|\mathbf{q}(t_0)\| < r \quad \Rightarrow \quad \|\mathbf{q}(t)\| < R \qquad \forall t \geq t_0. \tag{2.25}$$

3. *Otherwise, the equilibrium point* $\mathbf{0}$ *is* ***unstable.***

This definition means that we can keep the state in a ball of arbitrarily small radius $R(B_R)$ by starting the state trajectory in a ball of sufficiently small radius $r(B_r)$ the attractiveness definition given below, means that states close to $\mathbf{0}$ actually converges to $\mathbf{0}$ as time $t$ goes to infinity. More formally:

**Definition 12** *[SL91]*

1. *The equilibrium point* $\mathbf{0}$ *of the autonomous system (2.23) is* ***attractive*** *if*

$$\exists\ r > 0 \text{ such that } \|\mathbf{q}(0)\| < r \quad \Rightarrow \quad \|\mathbf{q}(t)\| \to \mathbf{0} \quad \text{as} \quad t \to \infty.$$

2. *The equilibrium point* $\mathbf{0}$ *of the nonautonomous system (2.22) is* ***attractive*** *at time* $t_0$ *if*

$$\exists\ r(t_0) > 0 \text{ such that } \|\mathbf{q}(t_0)\| < r(t_0) \quad \Rightarrow \quad \|\mathbf{q}(t)\| \to \mathbf{0} \quad \text{as} \quad t \to \infty.$$

Combining the Lyapunov stability and the attractiveness definitions the result is asymptotic stability. More formally:

**Definition 13** *The equilibrium point* $\mathbf{0}$ *of the system (2.22) is* ***asymptotically stable*** *if it is Lyapunov stable and attractive.*

The asymptotic stability requires that there exists an attractive region for every initial time $t_0$. The size of attractive region and the speed of the trajectory convergence may depend on the initial time $t_0$.

**Definition 14** *An equilibrium point which is Lyapunov stable but not attractive is called* ***marginally stable****.*

To ilustrate these definitions consider Figure (2.7) below.

However the concept asymptotic stability is still not sufficient in many engineering applications, i.e., know that a system will converge to the equilibrium point after a infinite time. It is important to note that the definitions of asymptotic stability do not quantify the rate of convergence. There is a strong form of stability which demands an exponential rate of convergence, i.e., the concept of exponential stability is needed. This concept can be used to estimate how fast the system trajectory approaches to $\mathbf{0}$.

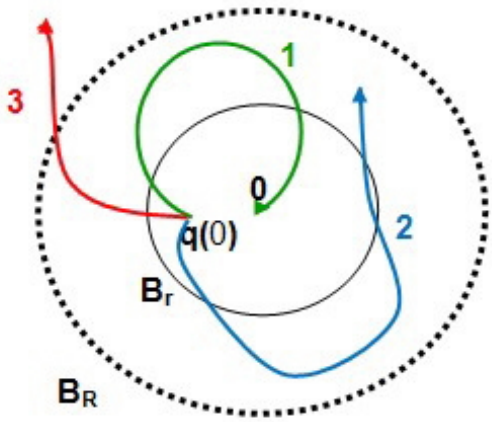


Figure 2.7: Concepts of stability [SL91]: curve 1 - asymptotically stable; curve 2 - marginally stable; curve 3 - unstable.

**Definition 15** *[SL91]*

1. *The equilibrium point* $\mathbf{0}$ *of the autonomous system (2.23) is* ***exponentially stable*** *if there exists two positive numbers,* $\alpha$ *and* $\lambda$*, such that for sufficiently small* $\mathbf{q}(t_0)$*,*

$$\|\mathbf{q}(t)\| \leq \alpha \|\mathbf{q}(0)\| e^{-\lambda t} \qquad \forall t > 0.$$

2. *The equilibrium point* $\mathbf{0}$ *of the nonautonomous system (2.22) is* ***exponentially stable*** *if there exists two positive numbers,* $\alpha$ *and* $\lambda$*, such that for sufficiently small* $\mathbf{q}(t_0)$*,*

$$\|\mathbf{q}(t)\| \leq \alpha \|\mathbf{q}(t_0)\| e^{-\lambda(t-t_0)} \qquad \forall t \geq t_0.$$

The above definitions are formulated to characterize the local behavior of systems (how the state evolves after starting near the equilibrium point). Global concepts are required to know how the system will behave when the initial state is some distance away from the equilibrium.

**Definition 16** *[SL91] If asymptotic (or exponential) stability holds for any initial states, the equilibrium point is said to be* ***globally asymptotically stable*** *(or* ***globally exponentially stable****).*

The concept, uniformity, is necessary to characterize nonautonomous systems whose behavior has a certain consistency for different values of initial time $t_0$. Uniform stability is a concept which guarantees that the equilibrium point is not losing stability. For a uniformly stable equilibrium point $\mathbf{q}^*$, $r$ in the Lyapunov stability definition should not be a function of $t_0$, so that equation (2.24) may hold for all $t_0$.

**Definition 17** *[SL91] The equilibrium point* $\mathbf{0}$ *of the system (2.22) is* ***uniformly stable*** *if, the saclar* $r$ *in the (2.24) can be chosen independently of* $t_0$*, i.e., if* $r = r(R)$.

**Definition 18** *[SL91] An equilibrium point* $\mathbf{0}$ *of the system (2.22) which is uniformly stable, is* ***uniformly asymptotically stable*** *if, there is a ball of attraction* $B_{R_0}$*, whose radius is independent of* $t_0$*, such that any system trajectory with initial states in* $B_{R_0}$ *converges to* $\mathbf{0}$ *uniformly in* $t_0$.

**Definition 19** *[SL91] The equilibrium point* $\mathbf{0}$ *of the system (2.22) is* ***globally uniformly asymptotically stable*** *if the uniformly stability holds for any initial states.*

### Lyapunov's linearization method

Consider the autonomous system in (2.23) with $\mathbf{f}(\mathbf{0}) = \mathbf{0}$, and assume that $\mathbf{f}$ is continuously differentiable. Using the Taylor expansion the system dynamics can be written as

$$\dot{\mathbf{q}} = \left(\frac{\partial \mathbf{f}}{\partial \mathbf{q}}\right)_{\mathbf{q}=\mathbf{0}} \mathbf{q} + \mathbf{h}(\mathbf{q}),$$

where $\mathbf{h}$ stands for higher-order terms in $\mathbf{q}$.

Let $\mathbf{A}$ be the Jacobian matrix of $\mathbf{f}$ with respect to $\mathbf{q}$ at $(\mathbf{q} = \mathbf{0})$ (an $n \times n$ matrix of elements $\frac{\partial f_i}{\partial q_j}$, $i, j = 1..., n$),

$$\mathbf{A} = \left(\frac{\partial \mathbf{f}}{\partial \mathbf{q}}\right)_{\mathbf{q}=\mathbf{0}},$$

so, the system

$$\dot{\mathbf{q}} = \mathbf{A}\ \mathbf{q}$$

is the *linearization* (or *linear approximation*) of the original nonlinear system at the equilibrium point $\mathbf{0}$.

The Lyapunov's linearization method is concerned with the small motion of nonlinear systems around equilibrium points, that is, it is concerned with the local stability of a nonlinear system.

**Theorem 3** *[SL91]* ***(Lyapunov's linearization method)***

- *If the linearized system is strictly stable (that is, if all eigenvalues of* $\mathbf{A}$ *are strictly in the left-half complex plane), then the equilibrium point is asymptotically stable (for the actual nonlinear system).*
- *If the linearized system is unstable (that is, if at least one eigenvalue of* $\mathbf{A}$ *is strictly in the right-half complex plane), then the equilibrium point is unstable (for the nonlinear system).*
- *If the linearized system is marginally stable (that is, all eigenvalues of* $\mathbf{A}$ *are in the left-half complex plane, but at least one of them is on the imaginary axis), then one cannot conclude anything from the linear approximation (the equilibrium point may be stable, asymptotically stable, or unstable for the nonlinear system).*

The proof of this theorem can be found in [SL91].

### Lyapunov analysis for autonomous systems

Lyapunov introduced a criterion for the stability of the system, a property whereby all the trajectories $\mathbf{q}(t)$ of the system tend to the origin. This criterion involves the existence of a certain function $V$, now known as a Lyapunov function. Consider the following definitions:

**Definition 20** *[SL91]*

1. *A scalar continuous function* $V(\mathbf{q})$ *is said to be* ***locally positive definite*** *if*

   - $V(\mathbf{0}) = 0$;
   - $\mathbf{q} \in B_{R_0}$ :
   $$\mathbf{q} \neq \mathbf{0} \quad \Rightarrow \quad \mathbf{V}(\mathbf{q}) > 0. \tag{2.26}$$

   *If the above propertys holds over the whole state space, then* $V(\mathbf{q})$ *is said to be* ***globally positive definite.***

2. *A scalar continuous function* $V(\mathbf{q})$ *is said to be* ***locally positive semi-definite*** *if*
   $$\mathbf{V}(\mathbf{q}) \geq 0 \;\; \textit{for} \;\; \mathbf{q} \in B_{R_0} \;\; \textit{and} \;\; \mathbf{q} \neq \mathbf{0}. \tag{2.27}$$

   *If the above property holds over the whole state space, then* $V(\mathbf{q})$ *is said to be* ***globally positive semi-definite.***

3. *If the condition (2.26) is replaced by* $\mathbf{V}(\mathbf{q}) < 0$, *then* $\mathbf{V}(\mathbf{q})$ *is said to be* ***locally negative definite****. If the condition (2.27) is replaced by* $\mathbf{V}(\mathbf{q}) \leq 0$, $\mathbf{V}(\mathbf{q})$ *is said to be* ***locally negative semi-definite****. (Similarly we define* ***globally negative definite*** *and* ***globally negative semi-definite****).*

The method of Lyapunov functions plays a central role in the study of the stability of control systems.

**Definition 21** *[SL91] If, in a ball* $B_{R_0}$ *the function* $V(\mathbf{q})$ *is positive definite and has continuous partial derivatives, and if its time derivative along any state trajectory of system* $\dot{\mathbf{q}} = \mathbf{f}(\mathbf{q})$ *is negative semi-definite, i.e.,*

$$\dot{\mathbf{V}}(\mathbf{q}) \leq 0$$

*then* $\mathbf{V}(\mathbf{q})$ *is said to be a* ***Lyapunov function*** *for the system* $\dot{\mathbf{q}} = \mathbf{f}(\mathbf{q})$.

**Theorem 4** *[SL91] (**Lyapunov Theorem for local stabillity**)*
*If in a ball* $B_{R_0}$, *there exists a scalar function* $V(\mathbf{q})$ *with continuous first partial derivatives such that:*
- $V(\mathbf{q})$ *is positive definite (locally in* $B_{R_0}$*);*
- $\dot{V}(\mathbf{q})$ *is negative semi-definite (locally in* $B_{R_0}$*);*

*then the equilibrium point* $\mathbf{0}$ *is stable. If, actually, the derivative* $\dot{V}(\mathbf{q})$ *is locally negative definite in* $B_{R_0}$, *then the stability is asymptotic.*

**Theorem 5** *[SL91] **(Lyapunov Theorem for global stability)** Assume that there exists a scalar function* $V$ *of the state* $\mathbf{q}$, *with continuous first order derivatives such that:*
- $V(\mathbf{q})$ *is positive definite;*
- $\dot{V}(\mathbf{q})$ *is negative definite;*
- $V(\mathbf{q}) \to \infty$ *as* $\|\mathbf{q}\| \to \infty$;

*then the equilibrium at the origin is globally asymptotically stable.*

The proof of these theorems can be found in [SL91].

### Lyapunov analysis for nonautonomous systems

**Definition 22** *A scalar time-varying function* $V(\mathbf{q}, t)$ *is* ***locally positive definite*** *if* $V(0, t) = 0$ *and there exists a time-variant positive definite function* $V_0(q)$ *such that*

$$\forall t \geq t_0, \quad V(\mathbf{q}, t) \geq V_0(q).$$

Thus, a time-variant function is locally positive definite if it dominates a time-variant locally positive definite function. Globally positive definite functions can be defined similarly.

**Definition 23** *A scalar time-varying function $V(\mathbf{q},t)$ is said to be* ***decrescent*** *if $V(0,t)=0$ , and if there exists a time-variant positive definite function $V_1(\mathbf{q})$ such that*

$$\forall t \geq 0, \quad V(\mathbf{q},t) \geq V_1(q).$$

In other words, a scalar function V (q,t) is decrescent if it is dominated by a time-invariant positive definite function.

**Definition 24** *If in a certain neighborhood of the equilibrium point, $V$ is positive definite and its derivative along the system trajectories is negative semi-definite, then $V$ is called* ***Lyapunov function for the non-autonomous system****.*

**Theorem 6** *[SL91] (**Lyapunov theorem for non-autonomous systems**)*

***Stability****: If, in a ball $B_{R_0}$ around the equilibrium point* $\mathbf{0}$*, there exists a scalar function $V(\mathbf{q},t)$ with continuous partial derivatives such that:*

*1. $V(\mathbf{q},t)$ is positive definite;*

*2. $\dot{V}(\mathbf{q},t)$ is negative semi-definite;*

*then the equilibrium point* $\mathbf{0}$ *is stable in the sense of Lyapunov.*

***Uniform stability and uniform asymptotic stability:*** *If, furthermore*

*3. $V(\mathbf{q},t)$ is decreasing;*

*then the origin is uniformly stable. If the condition 2 is strengthened by requiring $\dot{V}(\mathbf{q},t)$ to be negative definite, then the equilibrium point is asymptotically stable.*

***Global uniform asymptotic stability:*** *If, the ball $B_{R_0}$ is replaced by the whole state space, and condition 1, the strengthened condition 2, condition 3, and the condition*

*4. $V(\mathbf{q},t)$ is radially unbounded;*

*are all satisfied, then the equilibrium point at 0 is globally uniformly asymptotically stable.*

The prove of this theorem can be found in [SL91].

It is important to realize that the theorems in Lyapunov analysis are all sufficiency theorems. If for a particular choice of Lyapunov function candidate $V$, the conditions on $\dot{V}$ are not met, one cannot draw any conclusions on the stability or instability of the system - the only conclusion one may draw is that a different Lyapunov function candidate should be tried.

### La Salle´s Invariant set theory

Asymptotic stability of a control system is often an important property to be determined. Lyapunov's stability theorems studied above are often difficult to apply to establish this property, as it often happens that $\dot{V}$ (the derivative of the Lyapunov function candidate) is only negative semi-definite. In this kind of situation, it is still possible to draw conclusions on asymptotic stability, with the help of the invariant set theorems, which are attributed to La Salle.

**Definition 25** *[SL91] A set $S$ is an* ***invariant set*** *for a dynamic system if every trajectory which starts from a point in $S$ remains in $S$ for all time.*

The invariant set theorems reflect the intuition that the decrease of a Lyapunov function $V$ has to gradually vanish (i.e., has to converge to zero) because $\dot{V}$ is lower bounded.

**Theorem 7** *[SL91] (**Local Invariant set theorem**).*

*Consider an autonomous system of the form $\dot{\mathbf{q}} = \mathbf{f}(\mathbf{q})$, with $\mathbf{f}$ continuous and let $V(\mathbf{q})$ be a scalar function with continuous first partial derivatives. Assume that:*

- *for some $l > 0$, the region $\Omega_l$ defined by $V(\mathbf{q}) < l$ is bounded.*
- *$\dot{V}(\mathbf{q}) \leq 0$ for all $\mathbf{q}$ in $\Omega_l$.*

*Let $R$ be the set of all points within $\Omega_l$ where $\dot{V}(\mathbf{q}) = 0$ and $M$ be the largest invariant set in $R$ (i.e., $M$ is the union of all invariant sets within $R$). Then, every solution $\mathbf{q}(t)$ originating in $\Omega_l$ tends to $M$ as $t \to \infty$.*

**Corollary 8** *[SL91] (**La Salle's principle to establish asymptotic stability**)*

*Consider the autonomous system $\dot{\mathbf{q}} = \mathbf{f}(\mathbf{q})$, with $f$ continuous and let $V(\mathbf{q})$ be a scalar function with continuous partial derivatives. Assume that in a certain neighborhood $\Omega$ of the origin:*

- *$V(\mathbf{q})$ is locally positive definite;*
- *$\dot{V}$ is negative semi-definite;*
- *the set $R$ defined by $V(\mathbf{q}) = 0$ contains no trajectories of $\dot{\mathbf{q}} = \mathbf{f}(\mathbf{q})$ other than the trivial trajectory $\mathbf{q} = 0$.*

*Then, the equilibrium point $\mathbf{0}$ is asymptotically stable. Furthermore, the largest connected region of the form $\Omega_l$ (defined by $V(\mathbf{q}) < l$) within $\Omega$ is a domain of attraction of the equilibrium point.*

**Remark 1** *If in the above theorem $\Omega_l$ extends to the whole space $\mathbb{R}^n$ then global asymptotic stability can be established.*

The proofs of these results can be found in [SL91].

**Aplication: Lyapunov analysis of linear time-invariant systems** There is no general procedure for finding the Lyapunov functions for nonlinear systems, but for linear time invariant systems, the procedure comes down to the problem of solving a linear algebraic equation, called the *Lyapunov Algebraic Equation.*

**Definition 26** *A square matrix $M$ is* ***positive definite***[10] *if*

$$\mathbf{q} \neq \mathbf{0} \quad \Rightarrow \quad \mathbf{q}^T M \mathbf{q} > \mathbf{0},$$

*i.e., a matrix $M$ is positive definite if the quadratic function is a positive definite function.*

**Definition 27** *A square matrix $M$ is* ***semi-positive definite*** *if*

$$\forall \mathbf{q} \quad \Rightarrow \quad \mathbf{q}^T M \mathbf{q} \geq \mathbf{0}.$$

The concepts of negative definite and negative semi-definite can be defined similarly.

Given a linear system of the form $\dot{\mathbf{q}} = A\mathbf{q}$, let us consider a quadratic Lyapunov function candidate

$$V = \mathbf{q}^T P \mathbf{q},$$

where $P$ is a given symmetric positive definite matrix. Differentiating the positive definite function $V$ along the system trajectory yields another quadratic form:

$$\begin{aligned} \dot{V} &= \dot{\mathbf{q}}^T P \mathbf{q} + \mathbf{q}^T P \dot{\mathbf{q}} \\ &= \mathbf{q}^T A^T P \mathbf{q} + \mathbf{q}^T P A \mathbf{q} \\ &= -\mathbf{q}^T Q \mathbf{q}, \end{aligned}$$

where

$$A^T P + PA = -Q. \tag{2.29}$$

**Definition 28** *The matrix algebric equation (2.29) is called* ***Lyapunov Algebric Equation****.*

A more useful way of studying a given linear system using scalar quadratic functions is:

1. choose a positive definite matrix $Q$;
2. solve for $P$ from the Lyapunov equation (2.29);
3. check whether $P$ is positive definite.

[10] A positive definite matrix $M$ can always be decomposed as

$$M = U^T \Lambda U \tag{2.28}$$

where $U$ is a matrix of eigenvectors and satisfies $U^T U = I$, and $\Lambda$ is a diagonal matrix containing the eigenvalues of the matrix $M$. Let $\lambda_{\min}(M)$ denote the smallest eigenvalue of $M$ and $\lambda_{\max}(M)$ the largest one. Then, it follows from 2.28 that:

$$\lambda_{\min}(M) \|\mathbf{q}\|^2 \leq \mathbf{q}^T M \mathbf{q} \leq \lambda_{\max}(M) \|\mathbf{q}\|^2.$$

If $P$ is positive definite, then $\mathbf{q}^T P\mathbf{q}$ is a Lyapunov function for the linear system and global asymptotical stability is guaranteed.

Unlike the previous approach of going from a given $P$ to a matrix $Q$, this technique of going from a given $Q$ to a matrix $P$ always leads to conclusive results for stable linear systems, as seen from the following theorem.

**Theorem 9** *The linear time invariant system*

$$\dot{\mathbf{q}} = A\mathbf{q}$$

*is strictly stable if and only if for any symmetric positive definite matrix $Q$, the unique matrix $P$ solution of the Lyapunov equation (2.29) is symmetric positive definite.*

(See the proof of this result in [SL91]).

For the calculations to be computationally simple one often chooses $Q = I$.

### Barbalat´s Lemma

For autonomous systems, LaSalle's invariance set theorems allow asymptotic stability conclusions to be drawn even when $\dot{V}$ is only negative semi-definite in a domain $Q$. In that case, the system trajectory approaches the largest invariant set $S$, which is a subset of all points $\mathbf{q} \in Q$ where $\dot{V}(\mathbf{q}) = 0$. However the invariant set theorems are not applicable to nonautonomous systems. In this case, it may not even be clear how to define a set $S$, since $V$ may explicitly depend on both $t$ and $\mathbf{q}$.

**Lemma 10** ***(Barbalat Lemma)*** *Suppose $f(t)$ is differentiable and has a finit limit as $t \to \infty$. If $\dot{f}$ is uniformly continuous, then*

$$\lim_{t\to\infty} \dot{f}(t) = 0.$$

**Proof.** The proof is done by contradiction:

Suppose

$$\lim_{t\to+\infty} \dot{f}(t) \neq 0.$$

Then existe an $\epsilon > 0$ and a monotone increasing sequence $\{t_n\}$ such that:

$$\lim_{n\to+\infty} t_n = +\infty$$

and

$$\left|\dot{f}(t_n)\right| \geq \epsilon, \quad \forall n \in \mathbb{N}.$$

Since $f$ is uniformely continuous, for such $\epsilon$, then exists $\delta > 0$ such $\forall n \in \mathbb{N}$

$$|t - t_n| < \delta \Rightarrow \left|\dot{f}(t) - \dot{f}(t_n)\right| \leq \frac{\epsilon}{2}.$$

Hence if $t \in [t_n, t_n + \delta]$ then

$$\begin{array}{rcl} f'(t) & = & \left|\dot{f}(t_n) - \left(\dot{f}(t_n) - \dot{f}(t)\right)\right| \\ & \geq & \left|\dot{f}(t_n)\right| - \left|\dot{f}(t_n) - \dot{f}(t)\right| \\ & \geq & \epsilon - \frac{\epsilon}{2} \\ & = & \frac{\epsilon}{2}. \end{array}$$

Then, since $f \in C^1$, we have:

$$\begin{array}{rcl} \left|\int_a^{t_n+\delta} \dot{f}(t)\,dt - \int_a^{t_n} \dot{f}(t)\,dt\right| & = & \left|\int_{t_n}^{t_n+\delta} \dot{f}(t)\,dt\right| \\ & \geq & \int_a^{t_n+\delta} \left|\dot{f}(t)\right| dt \\ & \geq & \int_{t_n}^{t_n+\delta} \frac{\epsilon}{2} dt \\ & = & \frac{\epsilon\delta}{2} > 0. \end{array}$$

However,

$$\begin{array}{rcl} \lim\limits_{t\to\infty} \left|\int_a^{t_n+\delta} \dot{f}(t)\,dt - \int_a^{t_n} \dot{f}(t)\,dt\right| & = & \lim\limits_{t\to\infty} |f(t_n + \delta) - f(t_n)| \\ & = & \left|\lim\limits_{t\to\infty} f(t_n + \delta) - \lim\limits_{t\to\infty} f(t_n)\right| \\ & = & |\alpha| - |\alpha| = 0. \end{array}$$

This is a contradiction. Therefore

$$\lim_{t\to\infty} \dot{f}(t) = 0.$$

■

To the analysis of dynamic systems, one typically uses the following immediate corollary of the Barbalat's lemma to, which looks very much like an invariant set theorem in Lyapunov analysis:

**Lemma 11** *[SL91] ("**Lyapunov-Like Lemma**") If a scalar function $V(\mathbf{q}, t)$ satisfies the following conditions*

1. *$V(\mathbf{q}, t)$ is lower bounded;*

2. *$\dot{V}(\mathbf{q}, t)$ is negative semi-definite;*

3. *$\dot{V}(\mathbf{q},t)$ is uniformly continuous in time;*

*then $\dot{V}(\mathbf{q},t) \to 0$ as $t \to \infty$.*

**Proof.** Consider the hypotheses 1 and 2 we can conclude that $V(\mathbf{q},t)$ approaches a finite limiting value $\overline{v}$ such that $V(\mathbf{q},t) \geq \overline{v}$. Using the Barbalat´s Lemma and hypothese 3 we can conclude that

$$\dot{V}(\mathbf{q},t) \to 0 \quad \text{as} \quad t \to \infty.$$

■

Barbalat's lemma is a well-known and powerful tool to deduce asymptotic stability of nonlinear systems, especially time-varying systems, using Lyapunov like approaches (see [SL91] for a discussion and applications).

### II - Stability for control systems

The concept of stabilizability for the problem (2.13) is related to the existence of a feedback controller for a given system that makes the resulting system asymptotically stable about an equilibrium point.

**Definition 29** *[CLSS97] The system (2.13) is asymptotically controllable if:*

1. ***(attractiveness)*** *for each $\mathbf{q}_0 \in \mathbb{R}^n$ there exists some control such that the trajectory $\mathbf{q}(t) = \mathbf{q}(t;\mathbf{q}_0,\mathbf{u})$*[11] *is defined for all $t \geq 0$ and $\mathbf{q}(t) \to \mathbf{0}$ as $t \to \infty$;*

2. ***(Lyapunov stability)*** *for each $\varepsilon > 0$ there exists $\delta > 0$ such that for each $\mathbf{q}_0 \in \mathbb{R}^n$ with $|\mathbf{q}_0| < \delta$ there is a control $\mathbf{u}$ as in 1. such that $|\mathbf{q}(t)| < \varepsilon$ for all $t \geq 0$;*

3. ***(bounded controls)*** *there are a neighborhood $Q_0$ of $\mathbf{0}$ in $\mathbb{R}^n$, and a compact subset $U_0$ of $U$ such that, if the initial state $\mathbf{q}_0$ in 2. satisfies also $\mathbf{q}_0 \in Q_0$, then the control in 2. can be chosen with $\mathbf{u}(t) \in U_0$ for almost all $t$.*

When we were considering stability of dynamical systems, the fundamental definitions were given and the concept of Lyapunov stability was introduced. These concepts will now be extended to systems of the form (2.13) with a controlled input. Essentially, a Control Lyapunov Function can be defined as a positive definite (locally Lipschitz) function which becomes a Lyapunov function whenever a proper control action is applied.

A widely used technique for stabilization of this system to $\mathbf{q} = \mathbf{0}$ relies on the use of abstract "cost" functions that can be made to decrease in directions corresponding to possible controls. This approach is based on having a *Lyapunov pair* $(V, W)$:

[11] $\mathbf{q}(t;\mathbf{q}_0,\mathbf{u})$ denote the solution of (2.13) at time $t_0$, with initial condition $\mathbf{q}_0$ and control $\mathbf{u}$.

**Definition 30** *[SS95] A* ***control Lyapunov pair*** *for the system (2.13) consists of two continuous functions $V, W : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$, with $V$ continuously differentiable and $W$ continuous and such that the followimg properties hold:*

1. ***(positive definiteness)*** $V(\mathbf{0}) = 0$, *and* $\forall \mathbf{q} \neq \mathbf{0}, \quad \mathbf{V}(\mathbf{q}) > 0$ *and* $\mathbf{W}(\mathbf{q}) > 0$.
2. *(**properness**) the set*
$$\{\mathbf{q} : \quad \mathbf{V}(\mathbf{q}) \leq \beta\}$$
*is bounded for each $\beta$.*
3. *for each state $\boldsymbol{\xi} \in Q$ there is some control-value $\mathbf{u} = \mathbf{u}_\xi$ with*[12]
$$\nabla V(\boldsymbol{\xi}) \cdot \mathbf{f}(\boldsymbol{\xi}, \mathbf{u}) \leq -W(\boldsymbol{\xi}).$$

**Definition 31** *[SS95] A* ***control-Lyapunov function*** *(CLF) for the system (2.13) is a function $V : Q \to \mathbb{R}$ such that there exists a continuous positive definite $W : Q \to \mathbb{R}$ with the property that $(V, W)$ is a Lyapunov pair for (2.13).*

The process of designing a stable feedback controller is referred to, in control literature, as *feedback stabilization*. As pointed out in an early paper of Brockett (see the theorem below), some nonlinear systems cannot be stabilized by a continuous feedback. To illustrate this characteristic consider the Figure 2.8.

The vehicle is on the positive side of the $y$-axis, heading towards the left-half plane, i.e. $x = 0$, $y > 0$, and $\theta = \pi$. If $y \geq R_{\min}$, then it is possible to move towards the origin following a semi-circle. So, the decision in this case would be to turn left. On the other hand, if the car is too close to the origin, $y < R_{\min}$, then it needs to move away from the origin to obtain the correct heading. The decision in this case is to turn right. Therefore, the decision of where to turn to is a discontinuous function of the state, so is a discontinuous feedback.

Allowing discontinuous feedbacks is essential to stabilize some nonlinear systems, in particular the nonholonomic systems [Fon02a].

More precisely, this fact can be established rigorously, in 1983 in the Brockett's Necessary Condition [Bro83], as follows:

**Theorem 12** ***(Brockett's Necessary Condition)*** *[Bro83] Consider the system*
$$\dot{\mathbf{q}} = \mathbf{f}(\mathbf{q}, \mathbf{u}), \tag{2.30}$$
*with $\mathbf{f}(0,0) = 0$ and $\mathbf{f}(\cdot,\cdot)$ continuously differentiable in a neighborhood of the origin. If (2.30) is smoothly stabilizable, i.e., if there exists a continuously differentiable function $\mathbf{k}(\mathbf{q})$ such that the origin is an asymptotically stable equilibrium point of $\dot{\mathbf{q}} = \mathbf{f}(\mathbf{q}, \mathbf{k}(\mathbf{q}))$, with stability defined in the Lyapunov sense, then the image of $\mathbf{f}$ must contain an open neighborhood of the origin.*

[12] $\nabla V(\boldsymbol{\xi}) \cdot v$ is the directional derivative of $V$ in the direction of the vector $v$. This property guarantees that for each state $\boldsymbol{\xi}$ there is some control $\mathbf{u}(\cdot)$ such that, solving the initial-value problem (2.13) with $\mathbf{q}(0) = \boldsymbol{\xi}$, the resulting trajectory satisfies $\mathbf{q}(t) \to 0$ as $t \to +\infty$.

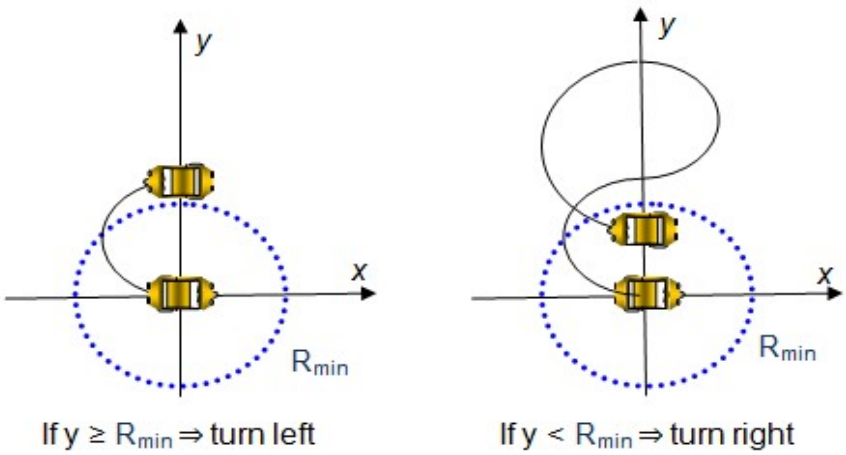


Figure 2.8: ([Fon02a]) A descontinuous decision.

This means that, if the system $\dot{\mathbf{q}} = \mathbf{f}(\mathbf{q}, \mathbf{u})$, with $\mathbf{u} \in U$, admits a continuos stabilizing feedback, then for any neighbourhood $V$ of zero, the set $\mathbf{f}(V, U)$ contains a neighbourd of zero.

However, if we allow discontinuous feedbacks, it might not be clear what is the solution of the dynamic differential equation. The trajectory $\mathbf{q}$, solution to

$$\dot{\mathbf{q}} = \mathbf{f}(\mathbf{q}(t), \mathbf{k}(\mathbf{q}(t))), \quad \mathbf{q}(0) = \mathbf{q}_0$$

is just defined, in a classical sense (known as Caratheodory solution), if $\mathbf{k}$ is assumed to be continuous. If $\mathbf{k}$ is discontinuous, a solution might even not exist! (See [Cla01] for a further discussion of this issue). A considerable effort has been expended [Ast96] in order to find the answer to the question

*"Is there a stabilizing feedback for every asymptotically controllable system?".*

One of the best-known candidates for the concept of solution of (2.13) is that of Filippov [CLSS97]. Filippov in 1988 [Fil88] define a solution to a dynamic equation with a discontinuous feedback law, the concept of Filippov solution, which has a number of desirable properties. However, it was shown by [Rya94] and by [CR94], that there are controllable systems, for example, nonholonomic systems, that cannot be stabilized, even allowing discontinuous feedbacks, if the trajectories are interpreted in a Filippov sense.

To illustrate the problem of defining a trajectory subject to a discontinuous feedback there is a known paradox [Fon02a]:

*Consider a runner, a cyclist, and a dog departing from the same point, at the same time, with the same directions, and with velocities $v$, $2v$ and $3v$ respectively. The runner and the cyclist*

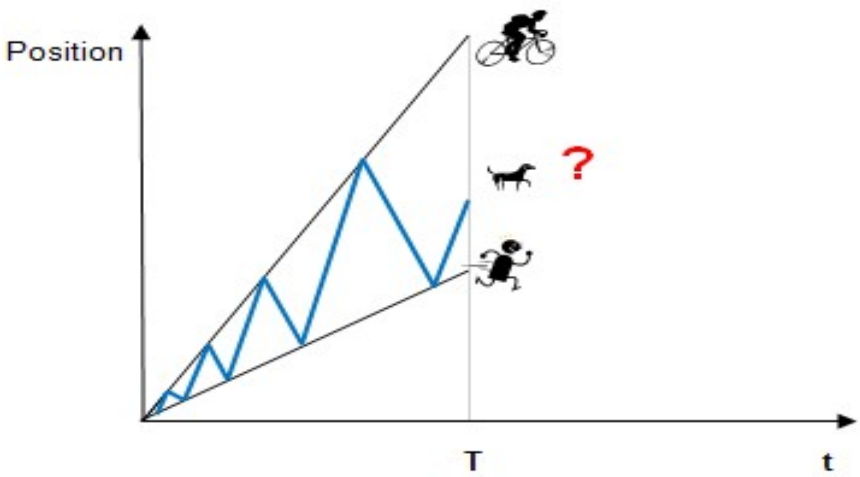


Figure 2.9: Undefined position of the dog.

*keep pursuing the same direction, while the dog uses the following "discontinuous strategy": it runs forward until it reaches the cyclist, then it turns and runs backward until it reaches the runner; at this point, it turns forward and repeats the procedure.*

**Question** *: where is the dog after a certain time $T$* (see Figure 2.9)*?*

**Answer:** *The dog can be at any point between the cyclist and the runner.*

*This might appear surprising since, apparently, every initial data is well defined and the final position of the dog is not. It is easy, however, to show that this is the answer if we reverse the time. Because, if the dog "starts" at time $T$ from anywhere between the cyclist and the runner, going backward in time, they will all meet at the origin at time zero.*

*A possible way out of this undefined situation is to force the dog to maintain its strategy for a certain time $\delta > 0$ before deciding where it should turn to the next time. The dog trajectory would then be well-defined and unique (see Figure 2.10). This idea of forcing a decision to be valid for a strictly positive time is used in the CLSS solution [CLSS97].*

Clarke, Ledyaev, Sontag and Subbotin in 1997 [CLSS97] proposed a solution concept that has been proved successful in dealing with stabilization by discontinuous feedbacks for a general class of controllable systems which is known by concept of **CLSS**. This concept use a strictly positive inter-sampling times. Consider a sequence of sampling instants

$$\pi := \{t_i\}_{i\geq 0}$$

in $[0, +\infty[$ with $t_0 < t_1 < t_2 < ...$ and such that $t_i \to \infty$ as $i \to \infty$. Let the function $t \mapsto \lfloor t \rfloor_\pi$ give the last sampling instant before $t$, that is

$$\lfloor t \rfloor_\pi := \max \{t_i \in \pi : t_i \leq t\}.$$

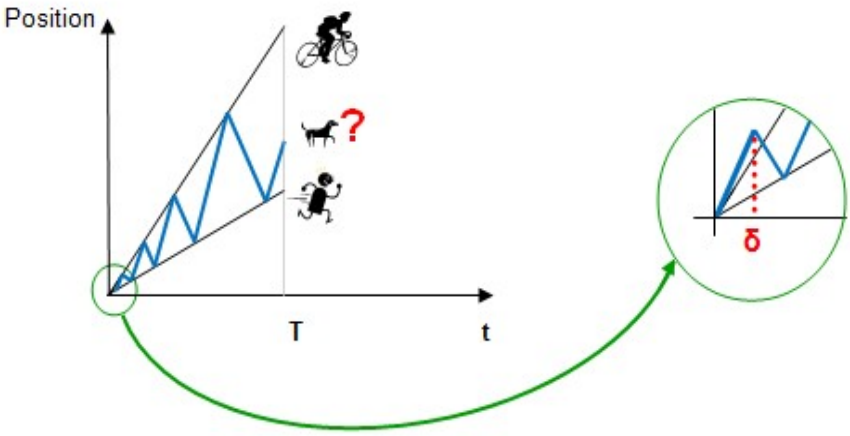


Figure 2.10: Well-defined position of the dog.

For a sequence $\pi$, the trajectory of the system under the feedback $\mathbf{k}$ is defined by

$$\begin{aligned} &\dot{\mathbf{q}}(t) = f\left(\mathbf{q}\left(t\right), \mathbf{k}\left(\mathbf{q}\left(\lfloor t \rfloor_{\pi}\right)\right)\right), \quad t \in [0, +\infty[\,, \\ &\mathbf{q}\left(t_0\right) = \mathbf{q}_0. \end{aligned} \tag{2.31}$$

Here, the feedback is not a function of the state at every instant of time, rather it is a function of the state at the last sampling instant.

With CLSS a longstanding open question in control theory was answered:

*"there is a stabilizing feedback for every asymptotically controllable system",*

this concept has been proved successful in dealing with stabilization by discontinuous feedbacks.

The traditional definition of asymptotical stability in the literature requires both attractiveness and Lyapunov stability to be satisfied, but attractiveness and Lyapunov stability are impossible to satisfy simultaneously for some systems that are controllable. One such system is the car-like vehicle (see application to the car-like).

Consider Figure 2.8. Lyapunov stability requires that the state stays arbitrarily close to the origin, provided the initial state is close enough to the origin. However, even if we start arbitrarily close to the origin, we might have to move to a certain minimal distance away from the origin in order to drive to it. Select an $\varepsilon > 0$ smaller that $R_{\min}$, and suppose the initial state is $\mathbf{q} = (x, y, \theta) = (0, \delta, 0)$. Even when selecting $\delta$ arbirarily small, the state must leave the $\varepsilon$-radius ball to drive to the origin. Therefore, to satisfy attractiveness, we cannot satisfy Lyapunov stability for this system.

Using the notion of stability in [CLSW98] which establishes just the usual notion of attractiveness, but not implying the Lyapunov stability, we have the following definition:

**Definition 32** *[Fon02a] The sampling-feedback* $\mathbf{k}$ *is said to* ***stabilize the system 2.31 on*** $Q_0$ *if there exists a sufficiently small inter-sample time* $\delta$ *such that the following condition is satisfied.*

*For any* $\gamma > 0$ *we can find a scalar* $M > 0$ *such that for any pair* $(t_0, \mathbf{q}_0) \in \mathbb{R} \times Q_0$ *we have*

$$\|\mathbf{q}(s + t_0; t_0, \mathbf{q}_0, \mathbf{k})\| < \gamma \quad \textit{for} \;\; s \geq M.$$

This definition of stability establishes just the usual notation of attractiveness, saying that

$$\|\mathbf{q}(t)\| \to 0 \;\; \text{as} \;\; t \to +\infty$$

does not imply the usual notion of Lyapunov stability,

$$\forall \varepsilon > 0 \quad \exists \delta > 0 : \quad \|\mathbf{q}(0)\| < \delta \Rightarrow \|\mathbf{q}(t)\| < \varepsilon.$$

## 2.3 Motion tasks

As already mentioned, the control of nonholonomic mechanical systems, such as mobile robots, has been the subject of a major research effort of the scientific community in recent years. If the aim is control problem for nonholonomic wheeled mobile robot moving on the plane, the possible motion tasks are [LOS98]:

- point stabilization;
- trajectory tracking;
- path-following.

In point stabilization we aim to drive the state (or position) of our system to a pre-specified point (usually the origin is considered, without loss of generality). A parking manoeuver is an example of such application in wheeled vehicles. In trajectory tracking we start at a given initial configuration and aim to follow as close as possible a trajectory in state space (i.e., a geometric path in the cartesian space together with an associated timing law). In path-following we just aim that the robot position follows a geometric path in the cartesian space, but no associated timing law is specified.

### 2.3.1 Point-to-point motion

In the point-to-point motion (point stabilization), the robot has to move from an initial to a final configuration in a given time $t_f$ (see Figure 2.11). The algorithm should find a control to generate a trajectory which is also capable to optimize some performance index when the point is moved from one position to another.

Finding a feedback control that solves the point-to-point motion task is a stabilization problem for an (equilibrium) point in the robot state space. For the differential-drive mobile robot car (see Figure 2.4), two control inputs are available for adjusting three configuration variables, namely the two cartesian coordinates characterizing the position of a reference point on the vehicle and its orientation.


Figure 2.11: Point-to-point motion.

As mensioned above, point-to-point stabilisation for nonholonomic vehicles cannot be achieved by a smooth feedback. The design of a feedback control law that drives the differential-drive mobile robot car (equation 2.17) to a desired configuration $\mathbf{q}_d$ will be done in the Chapter 6.

### 2.3.2 Trajectory tracking

In the trajectory tracking task, the robot must follow the desired Cartesian path with a specified timing law, i.e., the robot must track a moving reference robot. Although the trajectory can be split into a parameterized geometric path and a timing law for the parameter, such separation is not strictly necessary. Often, it is simpler to specify the workspace trajectory as the desired time evolution for the position of some representative point of the robot.

The trajectory tracking problem consists then in the stabilization to zero of the two-dimensional cartesian error $e_p = (e_1, e_2)$ (see Figure 2.12) using both control inputs.

Consider the differential-drive mobile robot car. Assume that the cartesian position in the plane of the midpoint of the axle connecting the rear wheels $(x, y)$ of the robot must follow the trajectory

$$(x_d(t), y_d(t)) \tag{2.32}$$

for $t \in [0, T]$.

The desired trajectory (equation 2.32) is feasible when it can be obtained from the evolution of a reference differential-drive mobile robot car, that is, the desired trajectory should be

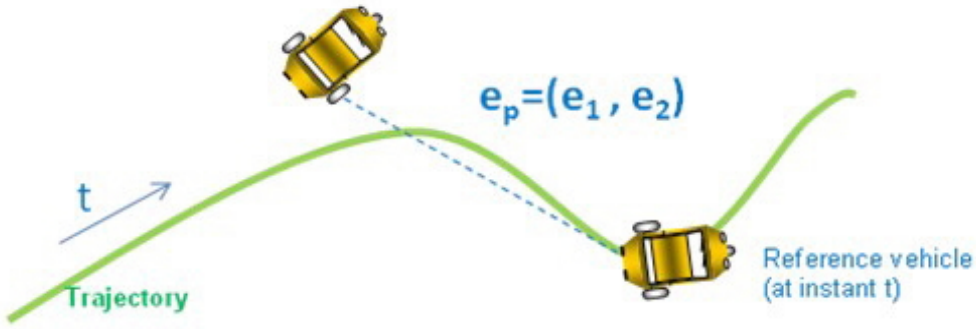


Figure 2.12: Trajectory tracking.

described by a reference state vector $(x_d(t), y_d(t), \theta_d(t))$ and a reference control signal vector $(v_d(t), w_d(t))$ and have the same constrains as given in equation (2.17) for $t \in [0, T]$

$$\dot{x}_d(t) = v_d(t) \cos \theta_d(t), \tag{2.33}$$

$$\dot{y}_d(t) = v_d(t) \sin \theta_d(t), \tag{2.34}$$

$$\dot{\theta}_d(t) = w_d(t), \tag{2.35}$$

for suitable initial conditions

$$(x_d(0),\ y_d(0),\ \theta_d(0))$$

and inputs

$$(v_d(t), w_d(t)).$$

Consider that $v_d(t) \neq 0$.

Solving for $v_d$ from equations (2.33) and (2.34) gives for the first input,

$$v_d(t) = \pm\sqrt{\dot{x}_d^2(t) + \dot{y}_d^2(t)}, \tag{2.36}$$

where the sign depends on the choice of executing the trajectory with forward or backward car motion, respectively.

From the kinematic model, equation (2.17) one has

$$\theta(t) = ATAN2\,(\dot{y}(t), \dot{x}(t)) + 2k\pi, \quad k \in \mathbb{Z}, \tag{2.37}$$

where $ATAN2$ is the four-quadrant inverse tangent function (undefined only if both arguments are zero).

Differentiating

$$\theta_d(t) = ATAN2\,(\ \dot{y}_d(t),\ \dot{x}_d(t))$$

with respect to time the second input $w_d(t)$ is computed.

Substituting the result in equation 2.35 yields the second input,

$$w_d(t) = \frac{\ddot{y}_d(t)\dot{x}_d(t) - \ddot{x}_d(t)\dot{y}_d(t)}{\dot{y}_d^2(t) + \dot{x}_d^2(t)}. \tag{2.38}$$

To reproduce the desired output trajectory the equations (2.36) and (2.38) provide the unique state and input trajectory. These expressions depend only on the values of the output trajectory, equation (2.32), and its derivatives up to the second order. Therefore, in order to guarantee its exact reproducibility, the cartesian trajectory should be two times differentiable in $[0, T]$.

Therefore, given an initial position $(x_d(0),\ y_d(0),\ \theta_d(0))^T$ and a consistent desired output trajectory $(x_d\,(t)\,, y_d\,(t))$ together with its derivatives, there is a unique associated state trajectory

$$\mathbf{q}_d(t) = (x_d(t), y_d(t), \theta_d(t))^T\,,$$

which can be computed in a purely algebraic way using

$$\theta_d(t) = ATAN2\ \ (\dot{y}_d(t), \dot{x}_d(t)) + 2k\pi, \quad k \in \mathbb{Z},$$

where the value of $k$ is chosen so that $\theta_d(0) = \theta(0)$. Hence, if needed by the tracking control scheme, the nominal orientation $\theta_d(t)$ may be computed off-line and used for defining a state trajectory error.

The state desired trajectory $\mathbf{q}_d(t) = (x_d(t), y_d(t), \theta_d(t))^T$ is the goal posture of the vehicle and the current state trajectory $\mathbf{q}(t) = (x(t), y(t), \theta(t))^T$ is the real posture at time $t$. Ideally, the errors $x_d - x,\ y_d - y$ and $\theta_d - \theta$ would be zero. But once $v_d(t)$ and $w_d(t)$ are open-loops controls, a feedback controller is needed to guarantee that these errors remain close to zero.

The trajectory tracking problem rather than using directly the state error

$$\mathbf{q}_d(t) - \mathbf{q}(t),$$

uses its rotated version defined as the state tracking error $\mathbf{e} = (e_1, e_2, e_3)^T$ as:

$$\begin{aligned} \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} &= \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_d - x \\ y_d - y \\ \theta_d - \theta \end{bmatrix} \\ &= \Re_\theta \begin{bmatrix} x_d - x \\ y_d - y \\ \theta_d - \theta \end{bmatrix}, \end{aligned} \tag{2.39}$$

where $(e_1, e_2)$ is $\mathbf{e}_p$ (see Figure 2.12) in a frame rotated by $\theta$ and the origen is $(x(t), y(t))$.

Since $\Re_\theta$ is invertible $\mathbf{e} \to \mathbf{0}$ implies that $\mathbf{q}(t) \to \mathbf{q}_d(t)$, which simply means that the difference to tracking objective has been achieved.

The aim now, is finding the kinematic model concerning the error posture $\mathbf{e}$.

**Lemma 13** *Define $v_d$ and $w_d$ as the desired linear and desired angular velocity of the tracked robot, respectivly. The kinematics of the tracking error posture can be described as:*

$$\dot{e}_1 = e_2 w - v + v_d \cos e_3, \tag{2.40}$$

$$\dot{e}_2 = -e_2 w + v_d \sin e_3, \tag{2.41}$$

$$\dot{e}_3 = w_d - w.$$

**Proof.** Using equation (2.39) and the nonholonomic constraint of the differential-drive mobile robot car

$$\dot{x}_d(t) \sin\theta_d(t) - \dot{y}_d(t) \cos\theta_d(t) = 0$$

we then have:

$$\begin{aligned}
\dot{e}_1 &= (\dot{x}_d - \dot{x}) \cos\theta - (x_d - x) \sin\theta\dot{\theta} + (\dot{y}_d - \dot{y}) \sin\theta + (y_d - y) \cos\theta\dot{\theta} \\
&= e_2\dot{\theta} - (\dot{x}\cos\theta + \dot{y}\sin\theta) + \dot{x}_d \cos\theta + \dot{y}_d \sin\theta \\
&= e_2 w - v + \dot{x}_d \cos(\theta_d - e_3) + \dot{y}_d \sin(\theta_d - e_3) \\
&= e_2 w - v + \dot{x}_d (\cos\theta_d \cos e_3 + \sin\theta_d \sin e_3) + \dot{y}_d (\sin\theta_d \cos e_3 - \cos\theta_d \sin e_3) \\
&= e_2 w - v + (\dot{x}_d \cos\theta_d + \dot{y}_d \sin\theta_d) \cos e_3 + (\dot{x}_d \sin\theta_d - \dot{y}_d \cos\theta_d) \sin e_3 \\
&= e_2 w - v + v_d \cos e_3,
\end{aligned}$$

$$\begin{aligned}
\dot{e}_2 &= -(\dot{x}_d - \dot{x}) \sin\theta + (x_d - x) \cos\theta\dot{\theta} + (\dot{y}_d - \dot{y}) \cos\theta - (y_d - y) \sin\theta\dot{\theta} \\
&= -e_1\dot{\theta} + \dot{x}\sin\theta - \dot{y}\cos\theta - \dot{x}_d \sin\theta + \dot{y}_d \sin\theta \\
&= -e_1 w - \dot{x}_d \sin(\theta_d - e_3) + \dot{y}_d \cos(\theta_d - e_3) \\
&= -e_1 w + \dot{x}_d (\sin\theta_d \cos e_3 - \cos\theta_d \sin e_3) + \dot{y}_d (\cos\theta_d \cos e_3 - \sin\theta_d \sin e_3) \\
&= -e_1 w + (\dot{x}_d \cos\theta_d + \dot{y}_d \sin\theta_d) \cos e_3 + (-\dot{x}_d \cos\theta_d + \dot{y}_d \sin\theta_d) \cos e_3 \\
&= -e_1 w + v_d \sin e_3,
\end{aligned}$$

$$\begin{aligned}
\dot{e}_3 &= \dot{\theta}_d - \dot{\theta} \\
&\quad\; w_d - w.
\end{aligned}$$

■

So,

$$\begin{bmatrix} \dot{e}_1 \\ \dot{e}_2 \\ \dot{e}_3 \end{bmatrix} = \begin{bmatrix} e_2 w - v + v_d \cos e_3 \\ -e_1 w + v_d \sin e_3 \\ w_d - w \end{bmatrix}. \tag{2.42}$$

Introducing the following change of velocity inputs [OLV02]:

$$\begin{aligned} u_1 &= -v + v_d \cos e_3, \\ u_2 &= w_d - w, \end{aligned}$$

then, the error state dynamic model (2.42) becomes,

$$\dot{\mathbf{e}} = \begin{bmatrix} o & w_d & 0 \\ -w_d & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{e} + \begin{bmatrix} 0 \\ \sin e_3 \\ 0 \end{bmatrix} v_d + \begin{bmatrix} 1 & -e_3 \\ 0 & e_1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}. \tag{2.43}$$

If we linearize the error dynamics around the reference trajectory ($e \approx 0$, hence $\sin e_3 \approx e_3$) we obtain

$$\dot{\mathbf{e}} = \begin{bmatrix} o & w_d & 0 \\ -w_d & 0 & v_d \\ 0 & 0 & 0 \end{bmatrix} \mathbf{e} + \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}. \tag{2.44}$$

The formulation of the tracking problem for the differential-drive mobile robot car is: find appropriate velocity controls laws $u_1$ and $u_2$ such that $\underset{t \to +\infty}{lim}\, \mathbf{e} = \mathbf{0}$.

**Lemma 14** *If $v_d$ and $w_d$ are nonzero constants the system (2.44) is controllable.*

**Proof.** When $v_d$ and $w_d$ are constant, the linear system (2.44) becomes time-invariant and controllable. Consider the matrix

$$C = \begin{bmatrix} B & AB & A^2B \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & -w_d^2 & w_d \\ 0 & 0 & -w_d & v_d & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

As $v_d$ and $w_d$ are nonzero constants $rank\,(C) = 3$, so using the necessary and sufficient condition for controllability (Kalman) The system (2.44) is controllable. ■

So[13],

$$\dot{\mathbf{e}} = \mathbf{A}\ \mathbf{e} + \mathbf{B}\ \mathbf{u}, \quad \text{with,} \quad \mathbf{u} = -\mathbf{K}\mathbf{e}$$

[13] Consider a nonautonomous nonlinear system with a control input $\mathbf{u}$.

$$\dot{\mathbf{q}} = \mathbf{f}\,(\mathbf{q}, \mathbf{u})$$

with $\mathbf{f}\,(\mathbf{0}, \mathbf{0}) = \mathbf{0}$, so, using the Taylor expansion, we can write:

$$\dot{\mathbf{q}} = \left(\frac{\partial \mathbf{f}}{\partial \mathbf{q}}\right)_{(\mathbf{q}=\mathbf{0},\mathbf{u}=\mathbf{0})} \mathbf{q} + \left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right)_{(\mathbf{q}=\mathbf{0},\mathbf{u}=\mathbf{0})} \mathbf{u} + \mathbf{h}\,(\mathbf{q}, \mathbf{u}),$$

where $\mathbf{h}$ stands for higher-order terms in $\mathbf{q}$ and $\mathbf{u}$.

Let $\mathbf{A}$ be Jacobian matrix of $\mathbf{f}$ with respect to $\mathbf{q}$ at $(\mathbf{q} = \mathbf{0}, \mathbf{u} = \mathbf{0})$ (an $n \times n$ matrix of elements $\frac{\partial f_i}{\partial q_j}$, $i, j = 1..., n$) and $\mathbf{B}$ be the Jacobian matrix of $\mathbf{f}$ with respect to $\mathbf{u}$ at $(\mathbf{q} = \mathbf{0}, \mathbf{u} = \mathbf{0})$ (an $n \times m$ matrix of elements

then,

$$\begin{aligned}\dot{\mathbf{e}} &= (\mathbf{A} - \mathbf{KB})\ \mathbf{e} \\ &= \overline{\mathbf{A}}\ \mathbf{e}.\end{aligned}$$

As the system is controllable, then it is stabilizable (that is, by an approppriate selection of $\mathbf{K}$, we can place the eigenvalues of $\overline{\mathbf{A}}$ with negative real part). Therefore we are in the conditions to apply Lyapunov's linearization method (Theorem 3).

### 2.3.3 Path following

In the path following task, the controller is given by a geometric description of the assigned cartesian path. This information is usually available in a parameterized form expressing the desired motion in terms of a path parameter $\Theta$, which may be, in particular, the arc length along the path. For this task, time dependence is not relevant because one is concerned only with the geometric displacement between the robot and the path. In this context, the time evolution of the path parameter is usually free and, accordingly, the command inputs can be arbitrarily scaled with respect to time without changing the resulting robot path.

The path following problem is thus rephrased as the stabilization to zero of a suitable scalar path error function (the distance $d$ to the path in Figure 2.13 ) using only one control input.

So, in the path following task a path (parametrized reference) $r(\Theta)$ is considered and is often given as a regular curve in the state space $\mathbb{R}^n$:

$$\begin{array}{rccc} r: & \left[\widehat{\Theta}, 0\right] \subset \mathbb{R} & \rightarrow & \mathbb{R}^n \\ & \Theta & \mapsto & r(\Theta), \end{array} \tag{2.45}$$

---

$b_{ij} = \frac{\partial f_i}{\partial u_j},\ i = 1...,n,\ j = 1...,m)$,

$$\mathbf{A} = \left(\frac{\partial \mathbf{f}}{\partial \mathbf{q}}\right)_{(\mathbf{q}=\mathbf{0},\mathbf{u}=\mathbf{0})} \quad \text{and} \quad \mathbf{B} = \left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right)_{(\mathbf{q}=\mathbf{0},\mathbf{u}=\mathbf{0})}.$$

So, the system

$$\dot{\mathbf{q}} = \mathbf{A}\ \mathbf{q} + \mathbf{B}\ \mathbf{u}$$

is the *linearization* (or *linear approximation*) of the original nonlinear system at $(\mathbf{q} = \mathbf{0}, \mathbf{u} = \mathbf{0})$.

Choosing a control law of the form $\mathbf{u} = \mathbf{u}(\mathbf{q})$, with $\mathbf{u}(\mathbf{0}) = \mathbf{0}$, and linearly approximating the control law as

$$\mathbf{u} = \left(\frac{d\mathbf{u}}{d\mathbf{q}}\right)_{\mathbf{q}=\mathbf{0}} \mathbf{q} = -\mathbf{K}\mathbf{q},$$

the closed-loop dynamics can be linearly approximated as

$$\dot{\mathbf{q}} = \mathbf{f}(\mathbf{q}, \mathbf{u}(\mathbf{q})) = (\mathbf{A} - \mathbf{BK})\ \mathbf{q}.$$

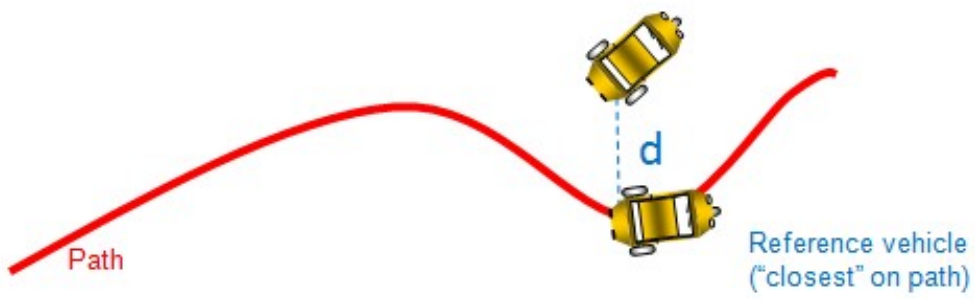


Figure 2.13: Path following.

$r(0) = 0$, and $r(\Theta) \in C^1$.

The path given above by (2.45) is negatively parametrized and ends in the origin. This path definition does not distinguish between finitely and infinitely long paths, since the real interval $\left[\widehat{\Theta}, 0\right]$ can be mapped onto both.

A path is regular if it is a non-singular curve and for each state $\mathbf{q} \in Q$ there exists a unique path parameter $\widetilde{\Theta}$, such that the distance between the path point $r(\widetilde{\Theta})$ and $\mathbf{q}$ is minimal. Combining equation (2.45) and with

$$e_T(t) = \mathbf{q}(t) - r(t)$$

yields to the path-following error:

$$e_P(t) = \mathbf{q}(t) - r(\Theta).$$

## 2.4 Notes at the end of chapter

Nonholonomic systems most commonly arise in finite dimensional mechanical systems where constraints are imposed on the motion that are not integrable, i.e. the constraints cannot be written as time derivatives of some function of the generalized coordinates. Such constraints can usually be expressed in terms of non integrable linear velocity relationships. Nonholonomic control systems result from formulations of nonholonomic systems that include control inputs. This class of nonlinear control systems has been studied by many researchers, and the published literature is now extensive. The interest in such nonlinear control problems is motivated by the fact that such problems are not amenable to methods of linear control theory, and they are not transformable into linear control problems in any meaningful way. Hence, these are nonlinear control problems that require fundamentally nonlinear approaches. On the other

hand, these nonlinear control problems are sufficiently special that good progress can be made. The literature that deals with the formulation of the equations of motion and the dynamics of nonholonomic systems is vast.

This Chapter serves as the support for the work developed. It doesn't contain original results. Here we studied the effect of nonholonomic constraints on the behavior of a robotic system.

The part of controllability (§ 2.2.1) was based on books [MLS94, SSVO09].

The book by Murray, Li, and Sastry [MLS94] provides a general introduction to nonholonomic control systems and present the study of nonholonomic constraints on the behavior of robotic systems. The book by Siciliano, Sciavicco, Villani and Oriolo [SSVO09] is recent (2009), and in it the concepts are introduced in a coherent and didactic way. Part of this book deal with wheeled mobile robots. It provides techiques for modelling, planning and control wheeled mobile robots.

The material on stability of nonholonomic systems (§ 2.2.2) was taken, mainly, from the book [SL91]. Many more details and properties can be found in this book.

Regarding on stability for control systems, the reference is the work of F. A. C. C. Fontes (see [Fon02a] and the references therein) and the work of Clark and all (see [CLSS97] and the references therein).

For the classification of possible motion tasks (§ 2.3), we used the work of A. De Luca, G. Oriolo and C. Samson [LOS98].

# Chapter 3

# Optimal Control

## 3.1 Introduction

From an modern point a view Optimal Control is a generalisation of the Calculus of Variations. For over three centuries, the Calculus of Variations played a central role in the development of mathematics and in the development of physics. In the 17th century, more precisely in 1696, Johann Bernoulli, challenged "the sharpest minds of the globe" with the Brachystochrone Problem:

"*find the path of shortest time of a particle moving between two points on a vertical plane*"

In the opinion of Sussmann and Willems [SW97] the *Brachystochrone Problem* marks the birth of Optimal Control. The Calculus of Variations was developed further in the 18th century by Euler and Lagrange and in the 19th century by Legendre, Jacobi, Hamilton, and Weierstrass. In the early 20th century, Bolza and Bliss gave the present rigorous mathematical structure on the subject. In the paper [SW97], we can find a historical perspective to the Optimal Control, from the Brachystochrone Problem to the Maximum Principle.

In general, it is accepted that the term "optimal control" came into use in the late 1950s to describe the problem of designing a controller to minimize a measure of a dynamical system's behavior over time. One of the approaches to this problem was developed by Richard Bellman and others in the middle of 1950s [Bel57]. This approach uses the concepts of a dynamical system's state and of a value function to define a functional equation, now often called the *Bellman equation.* The class of methods for solving optimal control problems by solving this equation came to be known as *Dynamic Programming.* In the 1960s, Kalman [Kal60] solved a problem with linear dynamics and integral quadratic cost function, showing that the optimal control is a linear feedback. This important instance of the optimal control problem was later named the *Linear Quadratic Regulator* (LQR). Also in 1962, a group of mathematicians L. S. Pontryagin, V. G. Boltyanskii, R. V. Gamkrelidze and E. F. Mishchenko, in the work of [PBGM62] solved Optimal Control problems relying on the Calculus of Variations. It is usually

accepted that the Maximum Principle was introduced by Pontryagin and his team in this paper. The Maximum Principle give necessary conditions of optimality for Optimal Control problems.

Before the 1950s, only rather simple problems could be solved, so, Calculus of Variations and Optimal Control were litle used by engineers [Bry96]. The truly enabling element for use of Optimal Control theory was the digital computer (which became available commercially in the 1950s) and with spectacular successes of optimal trajectory prediction in aerospace applications in the 1960s. In 1996 Bryson wrote a paper [Bry96] where we can find the history of Optimal Control since 1959 to 1985. When digital computers come into play, the Optimal Control theory was used to solve engineering problems, and according to [Bry96], to use digital computers for solving Optimal Control problems, one needs algorithms and reliable codes for these algorithms and this is perhaps the main difference between Optimal Control and Calculus of Variations.

A brief historical survey of development of the theory of Calculus of Variations and Optimal Control problems can be found in [Sar00].

From a very general way, the problem of Optimal Control is to find control functions so as to optimize a given performance index when subjected to constraints described by differential equations. The theory of Optimal Control is now a large area, with several possible approaches, with several areas of applicability, in mathematics, engineering, economics, social sciences, medicine, ... and a source of various techniques and ideas [Per01, Sar00].

## 3.2 The Optimal Control Problem

### 3.2.1 Model

In a control problem the system must be modelled. The objective is to obtain the mathematical description that adequately predicts the response of the physical system to all admissible controls. Consider a dynamical system modelled on some time interval $[t_0, t_f] \subset [0, +\infty[$ by a set of first order ordinary differential equations together with an initial condition and having the form:

$$\begin{cases} \dot{\mathbf{q}}(t) = f\left(t, \mathbf{q}(t)\right) & t \in [t_0, t_f]\,, \\ \mathbf{q}(t_0) = \mathbf{q}_0, \end{cases} \tag{3.1}$$

where, $t_0 \in [0, +\infty[$, the initial point, $\mathbf{q}_0 \in \mathbb{R}^n$ and the function $f : [t_0, +\infty[ \times \mathbb{R}^n \to \mathbb{R}^n$ are given. The unknown is the curve $\mathbf{q} : [t_0, +\infty[ \to \mathbb{R}^n$, which is interpreted as the dynamical evolution of the state of the system.

Suppose now that $f$ depends also on some "control" function $\mathbf{u}$ taking values on a set $U \subset \mathbb{R}^m$; so that $f : [t_0, +\infty[ \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$. Then, if the value of $\mathbf{u}\,(t) \in U$ is selected and if consider the corresponding dynamics:

$$\begin{cases} \dot{\mathbf{q}}(t) = f\left(t, \mathbf{q}(t), \mathbf{u}(t)\right) & t \in [t_0, t_f]\,, \\ \mathbf{q}(t_0) = \mathbf{q}_0, \end{cases} \tag{3.2}$$

we obtain the evolution of the system. The function $\mathbf{u} : [t_0, t_f] \to U$ is called a *control* or

*input* and is assumed that $\mathbf{u}$ is measurable and bounded. The vector $\mathbf{q}(t)$ represents the state variables, which characterize the behavior of the system at any time instant $t$.

The dynamics of the system, that is, the evolution of the state $\mathbf{q}(t)$ under a given control $\mathbf{u}(t)$, is assumed to be determined by:

$$\dot{\mathbf{q}}(t) = f(t, \mathbf{q}(t), \mathbf{u}(t)) \qquad t \in [t_0, t_f],$$

where

$$\mathbf{q}(t) = (q_1(t), ..., q_n(t))^T \quad \text{and} \quad \mathbf{u}(t) = (u_1(t), ..., u_m(t))^T.$$

Assuming also that: $f(t, \mathbf{q}(t), \mathbf{u}(t))$, $\partial f_i/\partial q_j$ and $\partial f_i/\partial u_k$ $(i, j = 1, ..., n;\ k = 1, ..., m)$, are all continuous on $[t_0, t_f] \times \mathbb{R}^n \times \mathbb{R}^m$. This assumption guarantees local existence and uniqueness of the solution of system (3.2) for a given $\mathbf{u} \in U$. Because $\mathbf{u}$ is measurable and bounded, the right side of the equation $\dot{\mathbf{q}}(t) = f(t, \mathbf{q}(t), \mathbf{u}(t))$ is continuous in $\mathbf{q}$ but only measurable and bounded in $t$ for each $\mathbf{q}$. Therefore, solutions are understood to be absolutely continuous functions that satisfy equation (3.2) almost everywhere [MS82].

### 3.2.2 Performance Criterion

The task is to determine what is the "best" control for the system. For this, one needs to specify a performance criterion for evaluating the performance of a system quantitatively.

So, we want to choose a measurable control function

$$\bar{\mathbf{u}} : [t_0, t_f] \to \mathbb{R}^m$$

and, thereby define the absolutely continuous trajectory

$$\bar{\mathbf{q}} : [t_0, t_f] \to \mathbb{R}^n,$$

in such a way that the pair $(\bar{\mathbf{q}}, \bar{\mathbf{u}})$ minimizes a given performance criterion represented by a functional of the type:

$$C_B(\mathbf{u}) := \int_{t_0}^{t_f} L(t, \mathbf{q}(t), \mathbf{u}(t))\ dt + W(\mathbf{q}(t_f)), \tag{3.3}$$

where $L : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ and $W : \mathbb{R}^n \to \mathbb{R}$ are given functions.

The functional (3.3) is the *objective function* (or *cost functional,* or simply *cost*) and can represent very general costs: running costs ($L$) like energy consumption or time spent, and terminal costs ($W$) like distance from target, among others.

**Definition 33** ***(Bolza form)*** *If the objective function is in the general form (3.3), it is said to be in the Bolza form, which corresponds to the sum of an integral term and a terminal term.*

Mayer and Lagrange problems are special Bolza problems with $L(t, \mathbf{q}(t), \mathbf{u}(t)) \equiv 0$ and $W(\mathbf{q}(t_f)) \equiv 0$, respectively.

**Definition 34** ***(Lagrange form)*** *If in the cost function (3.3) $W(\mathbf{q}(t_f)) \equiv 0$, then it may be defined in the so-called Lagrange form:*

$$C_L(\mathbf{u}) := \int_{t_0}^{t_f} L(t, \mathbf{q}(t), \mathbf{u}(t)) \; dt. \tag{3.4}$$

**Definition 35** ***(Mayer form)*** *If in the cost function (3.3) $L(t, \mathbf{q}(t), \mathbf{u}(t)) \equiv 0$, then it may be defined in the Mayer form:*

$$C_M(\mathbf{u}) := W(\mathbf{q}(t_f)). \tag{3.5}$$

The Mayer, Lagrange and Bolza problem formulations are theoretically equivalents [Cha07].

- **Lagrange problems can be reduced to Mayer problems** by introducing an additional state $q_e$, the new state vector $\tilde{\mathbf{q}} := (q_\varrho, q_1, ..., q_n)^T$ and an additional differential equation

  $$\dot{q}_\varrho(t) = L(t, \mathbf{q}(t), \mathbf{u}(t)); \qquad q_\varrho(t_0) = 0.$$

  Then, the cost functional (3.4) is transformed into one of the Mayer form (3.5) with

  $$W(\tilde{\mathbf{q}}(t_f)) := q_\varrho(t_f).$$

- **Mayer problems can be reduced to Lagrange problems** by introducing an additional state variable $q_e$, the new state $\tilde{\mathbf{q}} := (q_e, q_1, ..., q_n)^T$ and an additional differential equation

  $$\dot{q}_e(t) = 0; \qquad q_e(t_0) = \frac{1}{t_f - t_0} W(\mathbf{q}(t_f)).$$

  That is, the functional (3.5) can be rewritten in the Lagrange form (3.4) with:

  $$L(t, \tilde{\mathbf{q}}(t), \mathbf{u}(t)) := q_e(t).$$

- The previous transformations can be used to rewrite problems in Bolza form (3.3) in either the Mayer form or Lagrange form.

### 3.2.3 Constraints

In optimal control problems a great variety of constraints may be imposed. These constraints restrict the range of values that can be assumed by both the control and the state variables. Additional features can be added to the basic formulation of the Optimal Control problems to further expand the range of problems that can be addressed. For example:

- The initial state $\mathbf{q}_0$ to be chosen from a given set $Q_0$ instead of being fixed a priori;
- The problem is defined on an interval $[t_0, t_0+T]$ and $T$ is a decision variable (the *free-time problem versus the fixed-time problem*);
- Additional constraints can be added to the problem, for instance requiring the final state $\mathbf{q}(t_f)$ to be within a given set $S$ (*the constrained terminal state problem versus* the *free-terminal state problem* ($S = \mathbb{R}^n$));
- State constraints along the path can be represented by:
  - a functional equality
  $$k(t; \mathbf{q}(t)) = 0,$$
  for $t \in [t_0, t_f]$ and a given function $k : [t_0, t_f] \times \mathbb{R}^n \to \mathbb{R}$;
  - a functional inequality
  $$h(t; \mathbf{q}(t)) \leq 0,$$
  for $t \in [t_0, t_f]$ and a given function $h : [t_0, t_f] \times \mathbb{R}^n \to \mathbb{R}$.

The *point constraints* are used routinely in optimal control problems, especially terminal constraints (i.e., point constraints defined at terminal time).

An example, is an inequality terminal constraint of the form

$$h\left(t_f, \mathbf{q}\left(t_f\right)\right) \leq 0$$

that may apear in a stabilization problems, e.g., for forcing the system's response to belong to a given target set at terminal time.

The *path constraints* are encountered in many optimal control problems. Path constraints may be defined for restricting the range of values taken by mixed functions of both the control and the state variables. Moreover, such restrictions can be imposed over the entire time interval $[t_0, t_f]$ or any (nonempty) time subinterval, e.g., for safety reasons. For example, a path constraint could be define as

$$g\left(t, \mathbf{q}(t), \mathbf{u}(t)\right) \leq 0, \quad \forall t \in [t_0, t_f]$$

hence, restricting the points in phase space to a certain region $Q \subset \mathbb{R}^n$, $\forall t \in [t_0, t_f]$, that is, there exist forbidden zones of the state space.

**In conclusion:**

Particular cases of endpoint constraints can be consider:

· Initial state fixed and final state belonging to a set

$$\mathbf{q}(t_0) = \mathbf{q}_0 \quad \text{and} \quad \mathbf{q}(t_f) \in S.$$

· Equality endpoint constraints

$$h(\mathbf{q}(t_0)) = 0; \quad h(\mathbf{q}(t_0), \mathbf{q}(t_f)) = 0.$$

· Inequality endpoint constraints

$$h(\mathbf{q}(t_0)) \leq 0; \quad h(\mathbf{q}(t_0), \mathbf{q}(t_f)) \leq 0.$$

· Inclusion endpoint constraints

$$(\mathbf{q}(t_0), \mathbf{q}(t_f)) \in C_1 \times C_2, \quad \text{for given sets } C_1 \text{ and } C_2.$$

There are also other types of constraints that can be considered the "path constraints", theses appear in the form:

· Equality constraints

$$k(t, \mathbf{q}(t)) = 0, \quad \text{for} \quad t \in [t_0, t_f] \quad \text{and for a given function} \quad k : [t_0, t_f] \times Q \to \mathbb{R};$$

· Inequality constraints

$$h(t, \mathbf{q}(t)) \leq 0, \quad \text{for} \quad t \in [t_0, t_f] \quad \text{and for a given function} \quad h : [t_0, t_f] \times Q \to \mathbb{R};$$

· Implicit constraints

$$\mathbf{q}(t) \in C, \quad \text{for} \quad t \in [t_0, t_f] \quad \text{for a given set} \quad C.$$

These path constraints can be classificated, as:

· Pure constraints (constraints that depend on the trajectory), for example:

$$h(t, \mathbf{q}(t)) \leq 0, \quad \text{for} \quad t \in [t_0, t_f] \quad \text{and for a given function} \quad h : [t_0, t_f] \times Q \to \mathbb{R};$$

· Mixed constraints (constraints that depend on the trajectory and control), for example

$$h(t, \mathbf{q}(t), \mathbf{u}(t)) \leq 0 \quad \text{for } t \in [0, 1] \text{ a.e., and for a given function} \quad h : [t_0, t_f] \times Q \times U \to \mathbb{R}$$

**Example 4** *[Cha07] Consider a car parked at a given location $q_0 = q(t_0)$ and supose the aim is to drive the car to a pre-assigned destination $q(t_f) = q_f$ in a straight line, that is, the state heading towards to $(q_f, 0)$ (see Figure 3.1) and stop. For simplicity, the position $q(t)$ of the car is a unit point mass. The car can be decelerated or accelerated by using the brake or the accelerator, respectively. So, the control $u(t)$ is a real-valued function, representing the force on the car due to either decelerating $(u(t) \leq 0)$ or accelerating $(u(t) \geq 0)$ at time $t$. Supose, the control region $U$ is:*

$$U = \{u \in \mathbb{R} : u_{\min} \leq u(t) \leq u_{\max}\}.$$

*The aim is to know where the car is and how fast it is going.*

*The state is the vector $\mathbf{x}(t) = (q(t), \dot{q}(t))$ and, assuming that the car, at time $t = t_0$, is in position $q_0$, with velocity $v_0$, the initial state is $(q_0, v_0)$ .*

*The dynamics of the system are given by Newton's law $(F = ma)$ which can be writen as $\ddot{q}(t) = u(t)$. This equation in the vector form is:*

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t).$$

*The control problem is to bring the car at $q_f$ with velocity $0$, then a terminal constraint is imposed as*

$$\mathbf{x}(t_f) - \begin{bmatrix} q_f \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

*We can define a terminal set $S$ to be:*

$$S = \left\{(x_1, x_2) \in \mathbb{R}^2 : x_1 = q_f,\ x_2 = 0\right\}.$$

*Now, a selection of a performance measure is needed. For example:*

1. *Minimum time problems:*

   *Drive the state to some set $S$ in minimum time. The objective function is:*

   $$t_f - t_0 = \int_{t_0}^{t_f} 1\ dt$$

   *and the terminal constraint is $\mathbf{x}(q_f) \in S$.*

2. *Minimum energy / fuel problem:*

   *Drive the system into the set $S$ expending the minimum possible ammount of fuel.*

   *A typical objective function would be something like:*

   $$\int_0^{t_f} u^2(t)\ dt.$$

   *The time might be fixed or might be free.*

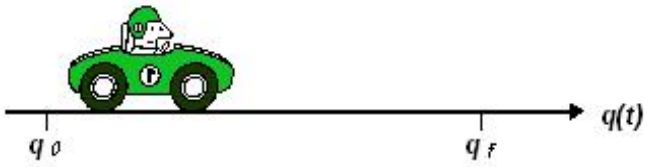


Figure 3.1: Control problem - simple car (adapted from [Cha07]).

### 3.2.4 Definition of the problem

The Optimal Control Problem consists of choosing a control function $\mathbf{u}(t) \in U$, thereby defining the trajectory $\mathbf{q}$, in such a way that the pair $(\mathbf{q}, \mathbf{u})$ minimises a given performance criterion

$$(P) \quad \begin{cases} \text{Minimize} & \int_0^T L(t, \mathbf{q}(t), \mathbf{u}(t))dt + W(\mathbf{q}(T)), \\ \text{subject to} & \dot{\mathbf{q}}(t) = f(t, \mathbf{q}(t), \mathbf{u}(t)), & \text{a.e. } t \in [0, T], \\ & \mathbf{q}(0) = \mathbf{q}_0 \in Q_0, \\ & \mathbf{q}(T) \in S, \\ & \mathbf{u}(t) \in U(t), & \text{a.e. } t \in [0, T]. \end{cases} \tag{3.6}$$

The data of this problem comprise:

- a terminal cost function $W : \mathbb{R}^n \to \mathbb{R}$, and a running cost function $L : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$;
- a function $f : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$;
- a set $Q_0 \subset \mathbb{R}^n$ containing all possible initial states at the initial time $t_0$;
- a multifunction[1] $U : \mathbb{R} \rightrightarrows \mathbb{R}^m$ of possible sets of control values;
- the terminal set $S \subset \mathbb{R}^n$.

A measurable function $u : [0, T] \to \mathbb{R}^m$ that satisfies $u(t) \in U$ *a.e.*, is called a *control function*. A *state trajectory* is an absolutely continuous function which satisfies $\dot{\mathbf{q}}(t) = f(t, \mathbf{q}(t), \mathbf{u}(t))$ for some control function. The domain of the optimization problem $(P)$ is the set of *admissible processes*, namely pairs $(\mathbf{q}, \mathbf{u})$, and comprises a control function $\mathbf{u}$ and corresponding state trajectory $\mathbf{q}$ which satisfy the constraints of $(P)$.

[1] [Vin00] Take a set $\Omega$. A *multifunction*

$$\Gamma : \Omega \rightrightarrows \mathbb{R}^n$$

is a mapping from $\Omega$ into the space of subsets of $\mathbb{R}^n$. For each $\omega \in \Omega$, then $\Gamma(\omega)$ is a subset of $\mathbb{R}^n$.

### 3.2.5 The open-loop *versus* the closed-loop Optimal Control Problem

**Definition 36 (Closed-loop Optimal control)** *[Cha07] If a functional relation of the form*

$$u^*(t) = w\left(t, q(t)\right)$$

*can be found for the optimal control at time $t$, then $w$ is called a closed-loop optimal control (or optimal feedback control) for the problem.*

**Definition 37 (Open-loop Optimal Control)** *[Cha07] The optimal control is said to be in open-loop form if the optimal control law is determined as a function of time for a specified initial state value, i.e.,*

$$u^*(t) = w\left(t, q(t_0)\right).$$

An open-loop optimal control is optimal only for a particular initial state value, whereas, if an optimal control law is known, the optimal control history can be generated from any initial state. An open-loop control acts based on the information on "how the model predicts the system to be".

The open-loop optimal controls are rarely applied directly in practice, it is not good for direct control due to the, for example, plant/model mismatch, process disturbance and initial condition error. The open-loop optimal controls can lead to infeasible operation due to constraints violation. However, the knowledge of an open-loop optimal control law for a given process can provide a good knowledge on how to improve system operation as well as some idea on how much can be gained upon optimization.

A closed-loop optimal control problem is solved for all (many) initial conditions. It is very difficult to find a solution and computationally very hard both in terms of time and memory. The closed-loop methods have as one of their main advantages precisely the fact that they provide a feedback control that acts based on the information on "how the system is". However, the price we pay to obtain the supreme goal the optimal feedback control is computationally very high. Moreover, open-loop optimal controls are routinely used in a number of feedback control algorithms such as Model Predictive Control. Model Predictive Control (MPC) generates a feedback law by solving a sequence of open-loop optimal control problems. At each sampling time $t_i$, an online open-loop optimal control problem is solved, using the current state $\mathbf{q}$ of the process as the initial state. The on-line optimization problem takes account of system dynamics, constraints, and control objectives. The optimization yields an optimal control sequence, and only the control action for the current time is applied while the rest of the calculated sequence is discarded. At the next time instant the horizon is shifted one sample period and the optimization is restarted with the information of the new measurements (see next Chapter).

## 3.3 Necessary Conditions of Optimality: Pontryagin Maximum Principle

A well-established tool for solving the open-loop Optimal Control Problem is necessary conditions in the form of a Maximum Principle. The Maximum Principle gives Necessary Conditions of Optimality (NCO) for Optimal Control Problems. It is usually accepted that the Maximum Principle was introduced by Pontryagin and his collaborators in 1962 [PBGM62].

Consider the problem stated in (3.6) in the fixed interval $[0,T]$, and assuming that $f$ and $L$ are differentiable, and $S$ is a convex set. The Maximum Principle states that if the pair $(\overline{\mathbf{q}},\overline{\mathbf{u}})$ is a minimizer, then there exists a scalar $\lambda \geq 0$ and an absolutely continuous function $\mathbf{p} : [0,T] \to \mathbb{R}^{1\times n}$, not simultaneously zero, such that the following conditions are satisfied:

- **The Adjoint Condition**:

$$-\dot{\mathbf{p}}(t) = \mathbf{p}(t) \cdot f_{\mathbf{q}}(t, \overline{\mathbf{q}}(t), \overline{\mathbf{u}}(t)) - \lambda L_{\mathbf{q}}(t, \overline{\mathbf{q}}(t), \overline{\mathbf{u}}(t)) \qquad a.e.;$$

- **The Transversality Condition**[2]:

$$-\mathbf{p}(T) - \lambda\, W_{\mathbf{q}}(\overline{\mathbf{q}}(T)) \;\in N_S(\overline{\mathbf{q}}(T));$$

- **The Maximization Condition (Weierstrass Condition)**:

  $\overline{\mathbf{u}}(t)$ maximizes

$$\mathbf{u} \to \mathbf{p}(t) \cdot f(t, \overline{\mathbf{q}}(t), \mathbf{u}) - \lambda L(t, \overline{\mathbf{q}}(t), \mathbf{u}) \qquad a.e.$$

**Remark 2** *(i) If $\mathbf{q} \in int\ S$ or $S = \mathbb{R}^n$, then $N_S(\mathbf{q}) = \{0\}$ and the Transversality Condition becomes*

$$-p(T) = \lambda\, W_{\mathbf{q}}(\overline{\mathbf{q}}(T)).$$

*(ii) If $S = \{0\}$, then $N_S(\mathbf{q}) = \mathbb{R}^n$ and the Transversality Condition disappears*

$$p(T) \in \mathbb{R}^n.$$

*In both cases we have two boundary conditions for the two differential equations:*

$$(i) \quad \begin{cases} \mathbf{q}(0) &= \mathbf{q}_0, \\ -p(T) &= \lambda\, W_{\mathbf{q}}(\mathbf{q}(t)), \end{cases} \qquad (ii) \quad \begin{cases} \mathbf{q}(0) &= \mathbf{q}_0, \\ \mathbf{q}(T) &= 0. \end{cases}$$

[2] [LFFdP11]Let S be a *convex set*, i.e., the line segment joining any two points of S belongs also to S.

A vector $\xi$ is a *vector normal* to the convex set S at the point $\overline{\mathbf{q}} \in S$, if $\xi$ do not make an acute angle with any segment in $S$ with extreme on $\overline{\mathbf{q}}$. That is,

$$\xi \cdot (\mathbf{q} - \overline{\mathbf{q}}) \leq 0, \quad \forall \mathbf{q} \in S.$$

The set of all vectors normal to $S$ at point $\overline{\mathbf{q}}$ is call *normal cone to $S$* at point $\overline{\mathbf{q}}$ and is represented by $N_S(\overline{\mathbf{q}})$,

$$N_S(\overline{\mathbf{q}}) = \{\xi \in \mathbb{R}^n : \xi \cdot (\mathbf{q} - \overline{\mathbf{q}}) \leq 0,\ \mathbf{q} \in S\}.$$

### 3.3.1 How to find a solution based on the NCO

Consider a problem with free terminal state, i.e., $S = \mathbb{R}^n$ and assume that the problem is normal, i.e., $\lambda = 1$.

A solution is obtained by solving a two point boundary-value problem[3], i.e., find $(\overline{\mathbf{q}}, p)$ subject to the ordinary differential equation and the boundary conditions:

$$\begin{aligned}
\dot{\overline{\mathbf{q}}}(t) &= f(t, \overline{\mathbf{q}}(t), \overline{\mathbf{u}}(t)) \qquad a.e.\ t \in [0, T], && (3.7)\\
-\dot{\mathbf{p}}(t) &= \mathbf{p}(t) \cdot f_{\mathbf{q}}(t, \overline{\mathbf{q}}(t), \overline{\mathbf{u}}(t)) - L_{\mathbf{q}}(t, \overline{\mathbf{q}}(t), \overline{\mathbf{u}}(t)) \qquad a.e., && (3.8)\\
\mathbf{q}(0) &= \mathbf{q}_0, && (3.9)\\
\mathbf{p}(T) &= -W_{\mathbf{q}}(\overline{\mathbf{q}}(T)), && (3.10)
\end{aligned}$$

with a pointwise maximization of $\mathbf{u}$ : $\overline{\mathbf{u}}$ maximizes

$$\mathbf{u} \to \mathbf{p}(t) \cdot f(t, \overline{\mathbf{q}}(t), \mathbf{u}) - L(t, \overline{\mathbf{q}}(t), \mathbf{u}). \tag{3.11}$$

**A (conceptual) algorithm**

1. Select an initial guess for the control function $\overline{\mathbf{u}}$.
2. Solve the two-point boundary value problem.
3. Check whether the maximization condition is verified.

   If YES, then STOP

   If NO, set the new control function $\overline{\mathbf{u}}$ to be the one solving the pointwise maximization.
4. Goto 2.

The problem of finding an actual pair of functions $(\overline{\mathbf{q}}, \overline{\mathbf{u}})$ satisfying the equations (3.7) to (3.11) above is known as the *two point boundary value problem* and its numerical solution is a well studied subject (see for example [K.E89]).

[3] If we define the (unmaximised) Hamiltonian as,

$$H(t, \mathbf{q}, \mathbf{p}, \mathbf{u}) = \mathbf{p}.f(t, \mathbf{q}, \mathbf{u}) - L(t, \mathbf{q}, \mathbf{u}),$$

the problem of finding an optimal solution can be restated as solving the Hamiltonian system of equations:

$$\begin{aligned}
\dot{\overline{\mathbf{q}}}(t) &= H_{\mathbf{p}}(t, \overline{\mathbf{q}}(t), \mathbf{p}(\mathbf{t}), \overline{\mathbf{u}}(t)),\\
-\dot{\mathbf{p}}(t) &= H_{\mathbf{q}}(t, \overline{\mathbf{q}}(t), \mathbf{p}(\mathbf{t}), \overline{\mathbf{u}}(t)),
\end{aligned}$$

with boundary conditions, $\mathbf{q}(0) = \mathbf{q}_0$ and $\mathbf{p}(T) = -W_{\mathbf{q}}(\overline{\mathbf{q}}(T))$ ,where $\overline{\mathbf{u}}(t)$ maximises over the function:

$$\mathbf{u} \longmapsto H(t, \overline{\mathbf{q}}(t), \mathbf{p}(\mathbf{t}), \mathbf{u})$$

## 3.4 Existence of Optimal Controls: deductive method

The application of NCO to identify a set of candidates to the optimal solution only makes sense if the optimal solution exists. If we do not verify whether, the solution to the problem exists, we might end up selecting an element from the set of candidates given by the NCO when this set does not contain the minimiser we seek. Therefore, there is great interest in studding the existence of optimal solutions. For Optimal Control Problems, results that guarantee the existence of solution can be found in [Cla83], for example. An essencial step before applying the NCO is the deductive method in optimization and it proceeds as follows [Cla89]:

1. A solution to the problem exists.
2. The necessary conditions are applicable, and they identify certain candidates to solution.
3. Further elimination, if necessary, identifies a solution.

Among several results on existence of solutions that can be found in the literature, we provide the following well-known result that will be especially useful later in the Chapter of Model Predictive Control. For a proof we refer to [Vin87].

**Theorem 15** *Assume that the data of problem $(P)$ satisfy:*

1. *The function $W$ is continuous.*
2. *The set $Q_0$ is compact and $S$ is closed.*
3. *There exists at least one admissible process.*
4. *The function $t \to f(t, \mathbf{q}, \mathbf{u})$ and $t \to L(t, \mathbf{q}, \mathbf{u})$ are measurable for all $(\mathbf{q}, \mathbf{u})$, and the functions $(\mathbf{q}, \mathbf{u}) \to f(t, \mathbf{q}, \mathbf{u})$ and $(\mathbf{q}, \mathbf{u}) \to L(t, \mathbf{q}, \mathbf{u})$ are continuous for all $t \in [0, T]$.*
5. *The function $\mathbf{q} \to f(t, \mathbf{q}, \mathbf{u})$ and $\mathbf{q} \to L(t, \mathbf{q}, \mathbf{u})$ are globally Lipschitz for all $t \in [0, T]$ and all $\mathbf{u} \in U(t)$ (with a Lipschitz constant $k$ neither depending on $t$ nor on $\mathbf{u}$.).*
6. *The "extended velocity set"*

$$\{(v, l) \in \mathbb{R}^n \times \mathbb{R}_+ : v = f(t, \mathbf{q}, \mathbf{u}), \quad l \geq L(t, \mathbf{q}, \mathbf{u}), \quad \mathbf{u} \in U(t)\}$$

*is convex for all $(t, \mathbf{q}) \in [0, T] \times \mathbb{R}^n$.*

*Then there exists an optimal process.*

## 3.5 Sufficient conditions and Dynamic Programming

Dynamic programming is a very powerful methodology in which a problem is solved by identifying a collection of subproblems and tackle them one by one, smallest first, using the answers to small problems to help the larger ones, until the whole lot of them is solved.

The main logic used to derive the Hamilton-Jacobi-Bellman (HJB) equation is the Principle of Optimality. To quote Bellman [Bel57]:

> "*An ptimal policy has the property that what ever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision.*"

For $\mathbf{q} \in \mathbb{R}^n$ and $0 \leq t_0 \leq T$ define the value function $V(t_0, \mathbf{q}(t_0))$ as the infimum cost from a initial pair time/state $(t_0, \mathbf{q}(t_0))$

$$V(t_0, \mathbf{q}(t_0)) = \inf_{\mathbf{u} \in U([t_0,T])} \left\{ \int_{t_0}^{T} L(s, \mathbf{q}(s), \mathbf{u}(s))\, ds + W(\mathbf{q}(T)) \right\}$$

**Theorem 16** ***(Hamilton-Jacobi Equation - HJE)*** *Assume that the value function $V$ is a $C^1$ function of the variables $(\mathbf{q}, t)$. Then $V$ solves the nonlinear partial differential equation,*

$$\begin{aligned} V_t(t, \mathbf{q}(t)) + \min_{\mathbf{u} \in U(t)} \{V_{\mathbf{q}}(t, \mathbf{q}(t)) \cdot f(t, \mathbf{q}(t), \mathbf{u}(t)) + L(t, \mathbf{q}(t), \mathbf{u}(t))\} &= 0 \qquad (3.12) \\ V(T, \mathbf{q}(T)) &= W(\mathbf{q}(T)). \end{aligned}$$

From the Principle of Optimality, we deduce that for any time subinterval $[t, t+\delta] \subset [t_0, T]$ $(\delta > 0)$ we have that for a trajectory $\mathbf{q}$ corresponding to $\mathbf{u}$

$$V(t, \mathbf{q}(t)) = \inf_{\mathbf{u} \in U([t,t+\delta])} \left\{ \int_{t}^{t+\delta} L(s, \mathbf{q}(s), \mathbf{u}(s))\, ds + V(t+\delta, \mathbf{q}(t+\delta)) \right\}, \qquad (3.13)$$

with boundary condition

$$V(T, \mathbf{q}(T)) = W(\mathbf{q}(T)).$$

Assuming the existence of a process $(\overline{\mathbf{q}}, \overline{\mathbf{u}})$ defined on $[t, t+\delta]$ which is actually a minimiser for the equation (3.13), we can write

$$-V(t, \overline{\mathbf{q}}(t)) + \int_{t}^{t+\delta} L(s, \overline{\mathbf{q}}(s), \overline{\mathbf{u}}(s))\, ds + V(t+\delta, \overline{\mathbf{q}}(t+\delta)) = 0$$

and for all pairs $(\mathbf{q}, \mathbf{u})$

$$-V(t, \mathbf{q}(t)) + \int_t^{t+\delta} L(s, \mathbf{q}(s), \mathbf{u}(s))\, ds + V(t+\delta, \mathbf{q}(t+\delta)) \geq 0. \tag{3.14}$$

Assume that $\overline{\mathbf{u}}$ and $\mathbf{u}$ are continuous from the right. Suppose also that $V$ is continuously differentiable and $L$ is continuous. We now want to convert this inequality into a differential form. So we rearrange (3.14) and divide by $\delta > 0$:

$$\frac{V(t+\delta, \mathbf{q}(t+\delta)) - V(t, \mathbf{q}(t))}{\delta} + \frac{1}{\delta}\int_t^{t+\delta} L(s, \mathbf{q}(s), \mathbf{u}(s))\, ds \geq 0$$

Let $\delta \to 0^+$, then we obtain the Hamilton-Jacobi Equation (3.12).

**Remark 3** *We can rewrite (3.12) as*

$$\begin{aligned} \textit{(HJE')} \qquad & V_t(t, \mathbf{q}) - \max_{\mathbf{u}\in\Omega(t)} H(t, \mathbf{q}, -V_{\mathbf{q}}(t, \mathbf{q}), \mathbf{u}) = 0 \\ & V(T, \mathbf{q}) = W(\mathbf{q}). \end{aligned} \tag{3.15}$$

*where,*

$$H(t, \mathbf{q}, p, \mathbf{u}) = p \cdot f(t, \mathbf{q}, \mathbf{u}) - L(t, \mathbf{q}, \mathbf{u})$$

Dynamic programming arguments lead us to the Hamilton-Jacobi equation, whose existence of a smooth solution provides a sufficient condition for optimality. The Hamilton-Jacobi equation provides just sufficient conditions of optimality (when these sufficient conditions are never satisfied, they will be of no help in finding the optimum).

## 3.6 Application: The Linear Quadratic Regulator

### The continuous-time Finite-Horizon LQR Problem

Consider the linear system

$$\dot{\mathbf{q}}(t) = A\, \mathbf{q}(t) + B\, \mathbf{u}(t) \quad \text{a.e.}, \tag{3.16}$$

where $\mathbf{q}(t) = (q_1(t), q_2(t), ..., q_n(t))^T$, are the components of the state vector $\mathbf{q}(t)$, $\mathbf{u}(t) = (u_1(t), u_2(t), ..., u_m(t))^T$ are the components of the control vector $\mathbf{u}(t)$. Thus, $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$.

Choose $S = \mathbb{R}^n$ and $U = \mathbb{R}^m$. The horizon $T$ can be an arbitrary positive number (finite horizon). The continuous LQR problem is to find the input function $\mathbf{u} : [0, T] \longrightarrow \mathbb{R}^m$ that minimizes the quadratic cost function

$$J_{cfin} := \int_0^T L(t, \mathbf{q}(t), \mathbf{u}(t))dt + W(\mathbf{q}(T)),$$

where

$$L(t, \mathbf{q}(t), \mathbf{u}(t))dt = \frac{1}{2}\left[\mathbf{q}^T(t)\ Q\ \mathbf{q}(t) + \mathbf{u}^T(t)\ R\ \mathbf{u}(t)\right]$$

and

$$W(\mathbf{q}(T)) = \mathbf{q}^T(T)\ M\ \mathbf{q}(T),$$

with $Q$ and $M$ symmetric positive definite matrices and with $R$ a positive definite matrix[4]. The problem is:

$$(P_{cfin}) \quad \begin{cases} \text{Minimize} & \frac{1}{2}\int_0^T \left[\mathbf{q}^T(t)\ Q\ \mathbf{q}(t) + \mathbf{u}^T(t)\ R\ \mathbf{u}(t)\right] dt + \mathbf{q}^T(T)\ M\ \mathbf{q}(T), \\ \text{subject to} & \dot{\mathbf{q}}(t) = A\ \mathbf{q}(t) + B\ \mathbf{u}(t), \qquad \text{a.e. } t \in [0, T], \\ & \mathbf{q}(0) = \mathbf{q}_0. \end{cases}$$

The solution to this problem exists (by Theorem 15) and the problem is normal (since there are no path nor terminal constraints).

The Hamiltonian is,

$$\begin{aligned} H(\mathbf{q}, \mathbf{p}, \mathbf{u}) &= \mathbf{p}^T(t).f(t, \mathbf{q}(t), \mathbf{u}(t)) - L(t, \mathbf{q}(t), \mathbf{u}(t)) \\ &= \mathbf{p}^T(t)(A\ \mathbf{q}(t) + B\ \mathbf{u}(t)) - \frac{1}{2}\left[\mathbf{q}^T(t)\ Q\ \mathbf{q}(t) + \mathbf{u}^T(t)\ R\ \mathbf{u}(t)\right]. \end{aligned}$$

If the pair $(\overline{\mathbf{q}}, \overline{\mathbf{u}})$ is a minimizer then the Adjoint Condition gives

$$\begin{aligned} -\dot{\mathbf{p}}^T(t) &= H_{\mathbf{q}}(t, \overline{\mathbf{q}}(t), \mathbf{p}(\mathbf{t}), \overline{\mathbf{u}}(t)) \\ &= \mathbf{p}^T(t) f_{\mathbf{q}}(t, \mathbf{q}(t), \mathbf{u}(t)) - L_{\mathbf{q}}(t, \mathbf{q}(t), \mathbf{u}(t)) \\ &= \mathbf{p}^T(t) A - \overline{\mathbf{q}}^T(t) Q, \end{aligned}$$

so,

$$-\dot{\mathbf{p}}(t) = A^T \mathbf{p}(t) - Q\ \overline{\mathbf{q}}(t). \tag{3.17}$$

From the Transversality Condition we get:

$$\mathbf{p}^T(T) = -\overline{\mathbf{q}}^T(t) M,$$

[4]Recall an $n \times n$ symmetric matrix $M$ is *positive definite* if

$$\mathbf{q}^T M \mathbf{q} > \mathbf{0}, \quad \text{for all } \mathbf{q} \neq 0,\ \mathbf{q} \in \mathbb{R}^n.$$

A symmetric matrix is positive definite if and only if all its eigenvalues are positive.

so,

$$\mathbf{p}\,(T) = -M\ \overline{\mathbf{q}}\,(t)\,. \tag{3.18}$$

Using the Maximization/Weierstrass Condition Principle, we get $\overline{\mathbf{u}}\,(t)$ that maximizes

$$\mathbf{u} \to H\,(\mathbf{q}, \mathbf{p}, \mathbf{u})\,.$$

As $H$ is a concave function of $\mathbf{u}$, the maximum is attained for points satisfying:

$$\nabla_{\mathbf{u}} H = 0.$$

So,

$$\mathbf{p}^T B - \mathbf{u}^T R = 0,$$

which is equivalent to

$$\mathbf{u} = R^{-1} B^T \mathbf{p}. \tag{3.19}$$

These conditions define the optimal solution. To obtain the optimal solution explicity we may solve a two point boundary value problem. Alternatively to this case, we may study the function $\mathbf{p}$.

Let

$$\mathbf{p}\,(t) = -P\,(t)\ \ \mathbf{q}\,(t)\,, \qquad \text{where}\ \ P : [0, T] \to \mathbb{R}^{n \times n}. \tag{3.20}$$

Using (3.18) and (3.20) we have:

$$P\,(T) = M,$$

and using (3.19) and (3.20) we have the feedback control[5]:

$$\mathbf{u}\,(t) = -R^{-1} B^T P\,(t)\ \ \mathbf{q}\,(t)\,. \tag{3.21}$$

We aim to determine $P(t)$ so that

$$-\dot{\mathbf{p}}\,(t) = \frac{d}{dt}\,(P\ \mathbf{q}) = P\,(t)\,\dot{\mathbf{q}}\,(t) + \dot{P}\,(t)\,\mathbf{q}\,(t)\,.$$

therefore, using (3.17) it becames

$$P\,(t)\,\dot{\mathbf{q}}\,(t) + \dot{P}\,(t)\,\mathbf{q}\,(t) = A^T \mathbf{p}\,(t) - Q\ \mathbf{q}\,(t)\,,$$

which is equivalent to:

$$P\,(t)\,(A\ \mathbf{q}(t) + B\ \mathbf{u}(t)) + \dot{P}\,(t)\,\mathbf{q}\,(t) = -A^T P\,(t)\ \ \mathbf{q}\,(t) - Q\ \mathbf{q}\,(t)\,.$$

Using (3.19) and (3.20) we obtain:

$$\left(PA - PBR^{-1}B^T P + \dot{P}\right)\mathbf{q} = \left(-A^T P - Q\right)\mathbf{q}.$$

[5]The assumption that $R$ is positive definite guarantees the existence of $R^{-1}$ $(\det R > 0)$.

This equation is satisfied if we can find $P(t)$ such that:

$$-\dot{P} = PA + A^T P - PBR^{-1}B^T P + Q \tag{3.22}$$

and

$$P(T) = M. \tag{3.23}$$

The equation (3.22) is called the *continuous-time Riccati equation.*

Suppose that $P$ is the symmetric and positive definite solution of the continuous-time Riccati equation and define the quadratic form

$$V(t) = \mathbf{q}^T(t)\, P(t)\, \mathbf{q}(t).$$

We note that

$$\begin{aligned}
\dot{V} &= \dot{\mathbf{q}}^T P\mathbf{q} + \mathbf{q}^T \dot{P}\mathbf{q} + \mathbf{q}^T P\dot{\mathbf{q}} \\
&= (A\mathbf{q} + B\mathbf{u})^T P\mathbf{q} + \mathbf{q}^T \dot{P}\mathbf{q} + \mathbf{q}^T P(A\mathbf{q} + B\mathbf{u}) \\
&= \mathbf{q}^T \left(A^T P + PA + \dot{P}\right)\mathbf{q} + \mathbf{u}^T B^T P\mathbf{q} + \mathbf{q}^T PB\mathbf{u}.
\end{aligned}$$

From the continuous-time Riccati equation (3.22), we have that

$$PA + A^T P + \dot{P} = PBR^{-1}B^T P - Q.$$

So,

$$\begin{aligned}
\dot{V} &= \mathbf{q}^T \left(PBR^{-1}B^T P\right)\mathbf{q} - \mathbf{q}^T Q\mathbf{q} + \mathbf{u}^T B^T P\mathbf{q} + \mathbf{q}^T PB\mathbf{u} + \mathbf{u}^T R\mathbf{u} - \mathbf{u}^T R\mathbf{u} \\
&= -\left(\mathbf{q}^T Q\mathbf{q} + \mathbf{u}^T R\mathbf{u}\right) + \left(B^T P\mathbf{q} + R\mathbf{u}\right)^T R^{-1}\left(B^T P\mathbf{q} + R\mathbf{u}\right).
\end{aligned}$$

Thus,

$$\int_0^T \dot{V}(t)\ dt = -2J_{cfin} + \int_0^T \left(B^T P\mathbf{q} + R\mathbf{u}\right)^T R^{-1}\left(B^T P\mathbf{q} + R\mathbf{u}\right)\ dt + \mathbf{q}^T(T)\ M\ \mathbf{q}(T)$$

$$\Leftrightarrow\ V(T) - V(0) = -2J_{cfin} + \int_0^T \left(B^T P\mathbf{q} + R\mathbf{u}\right)^T R^{-1}\left(B^T P\mathbf{q} + R\mathbf{u}\right)\ dt + \mathbf{q}^T(T)\ M\ \mathbf{q}(T) \tag{3.24}$$

$$\Leftrightarrow\ 2J_{cfin} = \mathbf{q}^T(0)\, P(0)\, \mathbf{q}(0) + \int_0^T \left(B^T P\mathbf{q} + R\mathbf{u}\right)^T R^{-1}\left(B^T P\mathbf{q} + R\mathbf{u}\right)\ dt. \tag{3.25}$$

The expression (3.24) is equivalent to expression (3.25), because:

$$V(T) = \mathbf{q}^T(T)\, P(T)\, \mathbf{q}(T) = \mathbf{q}^T(T)\, M\mathbf{q}(T).$$

Since the second term on the right-hand side is nonnegative, the minimum of $J_{cfin}$ is clearly achieved when

$$\mathbf{u} = R^{-1}B^T P(t)\,\mathbf{q}$$

and the minimum value of the cost is therefore given by

$$\min_{\mathbf{u}} J_{cfin} = \frac{1}{2}\mathbf{q}^T(0)\,P(0)\,\mathbf{q}(0)\,. \tag{3.26}$$

In summary, to solve the continuous-time finite-horizon LQR problem $(P_{cfin})$ we solve the continuous-time Riccati equation (3.22) with final condition (3.23) to determine $P(t)$ with $0 \le t \le T$ and set the optimal control as (3.21). The minimum value of the cost is therefore (3.26).

### The continuous-time Infinite-Horizon LQR Problem

Choosing $S = \mathbb{R}^n$, the horizon $T = \infty$ (infinite horizon). The continuous LQR problem is to find the input function $\mathbf{u}$ that minimizes the quadratic cost function

$$J_{c\inf} := \int_0^\infty L(t, \mathbf{q}(t), \mathbf{u}(t))dt,$$

where

$$L(t, \mathbf{q}(t), \mathbf{u}(t))dt = \frac{1}{2}\left[\mathbf{q}^T(t)\;Q\;\mathbf{q}(t) + \mathbf{u}^T(t)\;R\;\mathbf{u}(t)\right],$$

with $Q$ and $R$ symmetric positive definite matrices. The problem is:

$$(P_{c\inf})\quad \begin{cases} \text{Minimize} & \frac{1}{2}\int_0^\infty \left[\mathbf{q}^T(t)\;Q\;\mathbf{q}(t) + \mathbf{u}^T(t)\;R\;\mathbf{u}(t)\right] dt, \\ \text{subject to} & \dot{\mathbf{q}}(t) = A\;\mathbf{q}(t) + B\;\mathbf{u}(t), \qquad \text{a.e. } t \ge 0, \\ & \mathbf{q}(0) = \mathbf{q}_0. \end{cases}$$

So, if the pair $(\overline{\mathbf{q}},\overline{\mathbf{u}})$ is a minimizer then the Adjoint Condition gives:

$$-\dot{\mathbf{p}}(t) = A^T\mathbf{p}(t) - Q\;\overline{\mathbf{q}}(t)\,. \tag{3.27}$$

From the Transversality Condition we get:

$$\mathbf{p}(T) = \mathbf{0}. \tag{3.28}$$

Similarly to the previous case, we get:

$$\mathbf{u} = R^{-1}B^T\mathbf{p}. \tag{3.29}$$

Introducing $P : [0, T] \to \mathbb{R}^{n\times n}$ letting

$$\mathbf{p}(t) = -P(t)\ \mathbf{q}(t) \tag{3.30}$$

and using (3.28) and (3.30) we have:

$$P(T) = \mathbf{0}.$$

finaly, using (3.29) and (3.30) we have the feedback control:

$$\mathbf{u}(t) = -R^{-1}B^T P(t)\ \mathbf{q}(t). \tag{3.31}$$

This equation is satisfied if we can find $P(t) = P$ (that is, $P$ is constant) such that:

$$PA + A^T P - PBR^{-1}B^T P + Q = \mathbf{0}. \tag{3.32}$$

The equation (3.32) is called the *continuous-time Algebraic Riccati equation (ARE).*

Suppose that $P$ is the symmetric and positive definite solution of the Algebraic Riccati Equation (3.32) and define the quadratic form

$$V(t) = \mathbf{q}^T(t)\, P\mathbf{q}(t).$$

We note that

$$\begin{aligned}\dot{V} &= \dot{\mathbf{q}}^T P\mathbf{q} + \mathbf{q}^T P\dot{\mathbf{q}}\\ &= (A\mathbf{q} + B\mathbf{u})^T P\mathbf{q} + \mathbf{q}^T P(A\mathbf{q} + B\mathbf{u})\\ &= \mathbf{q}^T\left(A^T P + PA\right)\mathbf{q} + \mathbf{u}^T B^T P\mathbf{q} + \mathbf{q}^T PB\mathbf{u}.\end{aligned}$$

From the Algebraic Riccati Equation (3.32), we have that

$$PA + A^T P = PBR^{-1}B^T P - Q.$$

So,

$$\begin{aligned}\dot{V} &= \mathbf{q}^T\left(PBR^{-1}B^T P\right)\mathbf{q} - \mathbf{q}^{\mathbf{T}}Q\mathbf{q} + \mathbf{u}^{\mathbf{T}}B^T P\mathbf{q} + \mathbf{q}^{\mathbf{T}}PB\mathbf{u} + \mathbf{u}^{\mathbf{T}}R\mathbf{u} - \mathbf{u}^T R\mathbf{u}\\ &= -\left(\mathbf{q}^{\mathbf{T}}Q\mathbf{q} + \mathbf{u}^{\mathbf{T}}R\mathbf{u}\right) + \left(B^T P\mathbf{q} + R\mathbf{u}\right)^T R^{-1}\left(B^T P\mathbf{q} + R\mathbf{u}\right).\end{aligned}$$

Thus,

$$\int_0^\infty \dot{V}(t)\ dt = -2J_{c\inf} + \int_0^\infty \left(B^T P\mathbf{q} + R\mathbf{u}\right)^T R^{-1}\left(B^T P\mathbf{q} + R\mathbf{u}\right)\ dt$$

$$\Leftrightarrow\ \underbrace{V(\infty)}_{=0} - V(0) = -2J_{c\inf} + \int_0^\infty \left(B^T P\mathbf{q} + R\mathbf{u}\right)^T R^{-1}\left(B^T P\mathbf{q} + R\mathbf{u}\right)\ dt$$

$$\Leftrightarrow\ 2J_{c\inf} = \mathbf{q}^T(0)\, P\mathbf{q}(0) + \int_0^\infty \left(B^T P\mathbf{q} + R\mathbf{u}\right)^T R^{-1}\left(B^T P\mathbf{q} + R\mathbf{u}\right)\ dt.$$

Because the second term on the right-hand side is nonnegative, the minimum of $J_{c\inf}$ is clearly achieved when

$$\mathbf{u} = R^{-1}B^T P\mathbf{q},$$

and the minimum value of the cost is therefore

$$\min_{\mathbf{u}} J_{c\inf} = \frac{1}{2}\mathbf{q}^T(0)\,P\mathbf{q}(0)\,. \tag{3.33}$$

In summary, to solve the continuous-time infinite-horizon LQR problem ($P_{c\inf}$) we solve the continuous-time Riccati equation (3.32) with final condition (3.28) to determine $P$ and set the optimal control as (3.31). The minimum value of the cost is therefore (3.33).

**Remark 4** *Solving LQR by the Pontryagin Maximum Principle gives us linear Feedback control*

$$\mathbf{u} = K\ \mathbf{q}, \quad \textit{where} \quad K = -R^{-1}B^T P,$$

*but only if:*

1. *There are no inputs constraints ($U = \mathbb{R}^m$) nor terminal constraints ($S = \mathbb{R}^n$);*
2. *The objective function is exactly quadratic;*
3. *The dynamics are linear.*

*If one of these items is changed, then the Pontryagin Maximum Principle cannot give us feedback control directly. However, we can still have a linear (constant) feedback control for a finite-horizon problem, because if the terminal cost is conveniently selected, the finite and infinite problems are equivalent.*

*Suppose that we have a finite horizon LQR where the matrix $M$ of terminal cost is equal to $P$ solving the ARE. That is,*

$$J = \int_0^T \left(\mathbf{q}^T\mathbf{Q}\mathbf{q} + \mathbf{u}^T R\mathbf{u}\right)\ dt + \mathbf{q}^T(T)\,P\mathbf{q}(T)\,,$$

*where $P$ satisfies*

$$PA + A^T P - PBR^{-1}B^T P + Q = \mathbf{0}.$$

*It is easy to check that the constant matrix $P$ solves (3.22), (3.23) and also (3.32). So, the costs of the infinite and finite problems are equal*

$$\int_0^\infty \left(\mathbf{q}^T\mathbf{Q}\mathbf{q} + \mathbf{u}^T R\mathbf{u}\right)\ dt = \int_0^T \left(\mathbf{q}^T\mathbf{Q}\mathbf{q} + \mathbf{u}^T R\mathbf{u}\right)\ dt + \mathbf{q}^T(T)\,P\mathbf{q}(T)\,.$$

*The optimal controls are:*

$$\mathbf{u} = K\ \mathbf{q}, \quad \textit{where} \quad K = -R^{-1}B^T P.$$

*The problems are equivalent. This fact is important for the stability of the MPC (see quasi-infinite approach [CA98b]).*

### The discrete-time LQR Problem

To solve this problem numerically it is usual to consider the discrete-time version.

We are going to design the discrete linear-quadratic regulator for a continuous plant using [DL71].

Consider that a continuous system, such as (3.16) follows when $\mathbf{u}(t)$ is a piecewise constant function of time, i.e.,

$$\mathbf{u}(t) = \mathbf{u}_k, \quad t_k \leq t < t_{k+1} \quad \text{and} \quad k = 0, 1, ... \tag{3.34}$$

and that the state is sampled at the discrete time points $t_k$. The solution of (3.16) can be represented by[6]:

$$\mathbf{q}(t) = e^{At}\mathbf{q}(0) + \int_0^t e^{A(t-\tau)}\ B\ \mathbf{u}(\tau)\ d\tau.$$

Thus,

$$\begin{aligned}
\mathbf{q}(t+h) &= e^{A(t+h)}\mathbf{q}(0) + \int_0^{t+h} e^{A(t+h-\tau)}\ B\ \mathbf{u}(\tau)\ d\tau \\
&= e^{Ah}\left[e^{At}\mathbf{q}(0) + \int_0^t e^{A(t-\tau)}\ B\ \mathbf{u}(\tau)\ d\tau\right] + \int_t^{t+h} e^{A(t+h-\tau)}\ B\ \mathbf{u}(\tau)\ d\tau \\
&= e^{Ah}\mathbf{q}(t) + \int_t^{t+h} e^{A(t+h-\tau)}\ B\ \mathbf{u}(\tau)\ d\tau \\
&= \overline{\mathbf{A}}\ \mathbf{q}(t) + \overline{\mathbf{B}}\ \mathbf{u}(t).
\end{aligned}$$

[6] The solution to the linear differential equation (3.16) is given by:

$$\mathbf{q}(t) = e^{At}\mathbf{q}(0) + \int_0^t e^{A(t-\tau)}\ B\ \mathbf{u}(\tau)\ d\tau.$$

To prove this, we differentiate both sides. This gives:

$$\begin{aligned}
\frac{d\mathbf{q}}{dt} &= Ae^{At}\mathbf{q}(0) + \int_0^t Ae^{A(t-\tau)}\ B\ \mathbf{u}(\tau)\ d\tau + B\ \mathbf{u}(t) = \\
&= A\left[e^{At}\mathbf{q}(0) + \int_0^t e^{A(t-\tau)}\ B\ \mathbf{u}(\tau)\ d\tau\right] + B\ \mathbf{u}(t) = A\mathbf{q} + B\mathbf{u}.
\end{aligned}$$

Remember that $e^{At} = \sum_{k=0}^{\infty} \frac{1}{k!}\ A^k\ t^k$ and $\frac{d}{dt}e^{At} = A\ \sum_{k=0}^{\infty} \frac{1}{k!}\ A^k\ t^k = Ae^{At}$.

The behavior of the system at the sampling times $t = kh$ is described by the difference equation

$$\mathbf{q}_{k+1} = \overline{\mathbf{A}}\ \mathbf{q}_k + \overline{\mathbf{B}}\ \mathbf{u}_k\ , \quad 0 \leq k \leq N, \tag{3.35}$$

where $\mathbf{q}_k$ and $\mathbf{u}_k$ are, respectively, the state and the input vectors (note: for notational simplicity we represent $\mathbf{q}_k$ instead of $\mathbf{q}(k)$ and $\mathbf{u}_k$ instead of $\mathbf{u}(k)$). The relations between the system matrices $A$ and $B$ in continuous time and the representation (3.35) are as follows:

$$\overline{\mathbf{A}}(h) = e^{Ah}, \tag{3.36}$$

$$\overline{\mathbf{B}}(h) = \left(\int_0^h e^{A\tau}\ d\tau\right) B = Z(h)\ B. \tag{3.37}$$

in wich the matrix $Z(t)$ is defined as

$$Z(t) = \int_0^t e^{A\tau}\ d\tau.$$

Consider the linear quadratic regulator continuous-time cost

$$J_{c\inf 2} = \int_0^\infty \left[\mathbf{q}^T(t)\ Q\ \mathbf{q}(t) + \mathbf{u}^T(t)\ R\ \mathbf{u}(t)\right] dt,$$

i.e.,

$$J_{c\inf 2} = \sum_{k=0}^{\infty} \int_{t_k}^{t_{k+1}} \left[\mathbf{q}^T(t)\ Q\ \mathbf{q}(t) + \mathbf{u}^T(t)\ R\ \mathbf{u}(t)\right] dt.$$

Considering that, $h$ is constant in each interval $[t_k, t_{k+1}]$ following [PRMM10] and the addendum of this paper, we have:

$$\int_{t_k}^{t_{k+1}} \left[\mathbf{q}^T(t)\ Q\ \mathbf{q}(t) + \mathbf{u}^T(t)\ R\ \mathbf{u}(t)\right] dt = \mathbf{q}_k^T\ \overline{Q}_k\ \mathbf{q}_k + \mathbf{u}_k^T\ \overline{R}_k\ \mathbf{u}_k + \mathbf{q}_k^T\ \overline{M}_k\ \mathbf{u}_k, \tag{3.38}$$

with

$$\overline{Q}_k = \int_0^h \left(e^{At}\right)^T\ Q\ e^{At}\ d\tau,$$

$$\overline{R}_k = \int_0^h \left\{R + [Z(t)\,B]^T(h)\ Q\ [Z(t)\,B]\right\}\ d\tau,$$

$$\overline{M}_k = \int_0^h \left(e^{At}\right)^T\ Q\ \ Z(t)\,B\ d\tau.$$

From (3.38) follows that, given an infinite discrete-time input sequence $(\mathbf{u}_0, \mathbf{u}_1, ...)$, assuming that the continuous-time input $\mathbf{u}$ is defined in (3.34), then

$$J_{c\inf 2} = \sum_{k=0}^{\infty} \left( \mathbf{q}_k^T \ \overline{Q}_k \ \mathbf{q}_k + \mathbf{u}_k^T \ \overline{R}_k \ \mathbf{u}_k + \mathbf{q}_k^T \ \overline{M}_k \ \mathbf{u}_k \right).$$

Considering the following change of variables:

$$\begin{array}{rcl} \widetilde{\mathbf{u}}_k & \leftarrow & \mathbf{u}_k - \overline{R}_k^{-1} M \mathbf{q}_k, \\ \widetilde{A} & \leftarrow & \overline{A} - \overline{B}\overline{R}_k^{-1}\overline{M}^T, \\ \widetilde{Q} & \leftarrow & \overline{Q} - \overline{M}\overline{R}_k^{-1}\overline{M}^T, \end{array}$$

the following discrete-time LQR problem with mixed state-input terms:

$$\left| \begin{array}{ll} \underset{(\mathbf{u}_0,\mathbf{u}_1,...)}{\text{Minimize}} & \sum_{k=0}^{\infty} \left( \mathbf{q}_k^T \ \overline{Q} \ \mathbf{q}_k + \mathbf{u}_k^T \ \overline{R} \ \mathbf{u}_k + \mathbf{q}_k^T \ \overline{M} \ \mathbf{u}_k \right), \\ \text{subject to} & \mathbf{q}_{k+1} = \overline{\mathbf{A}} \ \mathbf{q}_k + \overline{\mathbf{B}} \ \mathbf{u}_k \ , \end{array} \right. \tag{3.39}$$

according to [PRMM10] (Lemma 6) is equivalent to the following discrete-time LQR problem without mixed state-input terms:

$$\left| \begin{array}{ll} \underset{(\mathbf{u}_0,\mathbf{u}_1,...)}{\text{Minimize}} & \sum_{k=0}^{\infty} \left( \mathbf{q}_k^T \ \widetilde{Q} \ \mathbf{q}_k + \widetilde{\mathbf{u}}_k^T \ \overline{R} \ \widetilde{\mathbf{u}}_k \right), \\ \text{subject to} & \mathbf{q}_{k+1} = \widetilde{\mathbf{A}} \ \mathbf{q}_k + \overline{\mathbf{B}} \ \widetilde{\mathbf{u}}_k \ . \end{array} \right. \tag{3.40}$$

Once again by using [PRMM10] (Lemma 7), the optimal cost-function value for the unconstrained discrete-time LQR problem (3.40) is

$$\mathbf{q}^T(0) \ \mathbf{\Pi} \ \mathbf{q}(0),$$

in which $\mathbf{\Pi}$ is the positive semi-definite solution of the Riccati equation:

$$-\mathbf{\Pi} + \widetilde{Q} + \widetilde{\mathbf{A}}^T \mathbf{\Pi} \widetilde{\mathbf{A}} - \widetilde{\mathbf{A}}^T \mathbf{\Pi} \overline{\mathbf{B}} \ \left( \overline{R} + \overline{\mathbf{B}}^T \mathbf{\Pi} \overline{\mathbf{B}} \right)^{-1} \overline{\mathbf{B}} \mathbf{\Pi} \widetilde{\mathbf{A}} = \mathbf{0}. \tag{3.41}$$

### The constrained LQR Problem

The work of A. Bemporad, M. Morari, V. Dua and E. Pistikopoulos in 2002 [BMDP02] showed how to compute the solution to the constrained optimal control problem as a piecewise affine state-feedback law. Such a law is computed off-line by using a multi-parametric programming solver [BMDP02, BMDP02], which divides the state space into polyhedral regions, and for each region determines the affine gain which produces the optimal control action.

Consider the discrete-time linear time-invariant system

$$\mathbf{q}(t+1) = \mathbf{A}\mathbf{q}(t) + \mathbf{B}\mathbf{u}(t)$$

and a linear state feedback control law that regulate the state to the origin

$$\mathbf{u}(t) = K\mathbf{q}(t),$$

where $\mathbf{q} \in \mathbb{R}^n$, $\mathbf{u} \in \mathbb{R}^m$, $\mathbf{A} \in \mathbb{R}^{n\times n}$, $\mathbf{B} \in \mathbb{R}^{n\times m}$ and $K$ is a constant matrix gain. Suppose that it is required that the closed loop system satisfies the output and input constraints:

$$\begin{aligned} \mathbf{q}_{\min} &\leq \mathbf{q}(t) \leq \mathbf{q}_{\max}, \\ \mathbf{u}_{\min} &\leq \mathbf{u}(t) \leq \mathbf{u}_{\max}, \end{aligned}$$

for all time $t \geq 0$, where $\mathbf{q}_{\min}$, $\mathbf{q}_{\max}$, $\mathbf{u}_{\min}$ and $\mathbf{u}_{\max}$ are are constant vectors of suitable dimension.

Let $\mathbf{q}(0) = \mathbf{q}_0$ be the initial state and consider the constrained finite time optimal control problem:

$$\left|\begin{array}{lll} \underset{(\mathbf{u}_0,\mathbf{u}_1,\ldots,\mathbf{u}_{N-1})}{\text{Minimize}} & \sum_{k=0}^{N-1}\left(\mathbf{q}_k^T\ Q\ \mathbf{q}_k + \mathbf{u}_k^T\ R\ \mathbf{u}_k\right) + \mathbf{q}_N^T\ Q\ \mathbf{q}_N, & \\ \text{subject to} & \mathbf{q}_{k+1} = \mathbf{A}\ \mathbf{q}_k + \mathbf{B}\ \mathbf{u}_k\ , & k = 0,1,\ldots,N-1 \\ & \mathbf{q}(0) = \mathbf{q}_0, & \\ & \mathbf{q}_{\min} \leq \mathbf{q}_k \leq \mathbf{q}_{\max}, & k = 0,1,\ldots,N-1 \\ & \mathbf{u}_{\min} \leq \mathbf{u}_k \leq \mathbf{u}_{\max}, & k = 0,1,\ldots,N-1 \\ & \mathbf{u}_{k+1} = K\mathbf{q}_{k+1}, & \end{array}\right. \tag{3.42}$$

with $Q$ and $R$ symmetric positive definite matrices and with $Q$ positive definite matrice.

The solution to problem (3.42) has been studied by Bemporad et.al. in [BMDP02]. We will briefly summarize the main results. The optimization problem (3.42) can be translated into a quadratic program (QP). By substituting

$$\mathbf{q}_k = \mathbf{A}^k\mathbf{q}_0 + \sum_{j=0}^{k-1}\mathbf{A}^j\mathbf{B}\mathbf{u}_{k-1}.$$

The problem (3.42) can be reformulated as

$$\begin{array}{ll} \underset{(\mathbf{u}_0,\mathbf{u}_1,\ldots,\mathbf{u}_{N-1})}{\text{Minimize}} & \left\{\frac{1}{2}\mathbf{U}^T\mathbf{H}\mathbf{U} + \mathbf{q}_0^T\mathbf{F}\mathbf{U}\right\} + \frac{1}{2}\mathbf{q}_0^T\mathbf{Y}\mathbf{q}_0, \\ \text{subject to} & \mathbf{G}\mathbf{U} \leq \mathbf{W} + \mathbf{E}\mathbf{q}_0, \end{array} \tag{3.43}$$

where the column vector

$$\mathbf{U} := \left[\mathbf{u}_0^T, \mathbf{u}_1^T, \ldots, \mathbf{u}_{N-1}^T\right]^T \in \mathbb{R}^{\overline{m}}, \qquad \overline{m} := m \times N$$

is the optimization vector, $\mathbf{H}$ is symmetric positive definite matrice, and $\mathbf{H}$, $\mathbf{F}$, $\mathbf{Y}$, $\mathbf{G}$, $\mathbf{W}$, and $\mathbf{E}$ are obtained from $Q$, $R$ and (3.42) (see [BMDP02] for details).

Denoting by $X_f \subset \mathbb{R}^n$ the set of initial states $\mathbf{q}$ for which the optimal control problem (3.42) is feasible, that is,

$$X_f = \left\{\mathbf{q} \in \mathbb{R}^n : \quad \exists U \in \mathbb{R}^{\overline{m}}, \quad \mathbf{GU} \leq \mathbf{W} + \mathbf{Eq}\right\}$$

and with $U^*(\mathbf{q})$ the optimizer of (3.43) for $\mathbf{q}_0$.

The problem (3.43) depends on $\mathbf{q}_0$, therefore the implementation of finite time constrained linear quadratic regulator can be performed by solving problem (3.43) for all $\mathbf{q}_0$ within a given range of values, that is, by considering problem (3.43) as a Multi-parametric Quadratic Program. We have the following result on the solution of (3.42):

**Theorem 17** *Consider the finite time constrained linear quadratic regulator (3.42). Then, the set of feasible parameters $X_f$ is convex, the optimizer $U : X_f \to \mathbb{R}^{\overline{m}}$ is continuous and piecewise affine, that is,*

$$U^*(\mathbf{q}_0) = F_k\mathbf{q}_0 + G_k, \qquad \forall \mathbf{q} \in P_k. \qquad k = 1, ..., N_p,$$

*where $F_k \in \mathbb{R}^{m\times n}$, $G_k \in \mathbb{R}^m$ and $P_k = \left\{\mathbf{q} \in \mathbb{R}^n : \ \overline{H}_k\mathbf{q} \leq H_k\right\}$, $\overline{H}_k \in \mathbb{R}^{p_k\times n}$, $H_k \in \mathbb{R}^{p_k}$, $k = 1, ..., N_p$ is a polyhedral partition[7] of $X_f$.*

In conclusion, the optimal control problem of a constrained linear discrete-time system can now be formulated as a multi-parametric quadratic program by treating the state vector as a parameter. The optimal solution is a piecewise affine state-feedback control law that is defined over a polyhedral partition of the feasible state-space.

**Remark 5** *Since the problem depends on $\mathbf{q}$, the implementation of MPC can be performed by solving problem (3.43) off-line for all $\mathbf{q}$ within a given range of values, that is, by considering (3.43) as a multi-parametric Quadratic Program (mp-QP).*

---

7

**Definition 38** (***Polyhedron***). *A convex set $S \subseteq \mathbb{R}^n$ given as an intersection of a finite number of closed half-spaces*

$$S = \{\mathbf{q} \in \mathbb{R}^n : \mathbf{Aq} \leq B\},$$

*where $\mathbf{A} \in \mathbb{R}^{p\times n}$, $\mathbf{B} \in \mathbb{R}^q$, is called polyhedrom.*

**Definition 39** (***Polytope***). *A bounded polyhedrom $\mathbf{P} \subset \mathbb{R}^n$ is called a polytope.*

**Definition 40** (***Partition***). *A collection of sets $\{P_k\}_{k=1}^N$ is a partition of a set $P$ if:*
*(i) $P = \bigcup_{k=1}^{N} P_k$ and (ii) $P_i \cap P_j = \varnothing \ \ \forall i \neq j$, with $i, j \in \{1, ..., N\}$.*

**Definition 41** (***Polyhedral Partition***). *A collection of sets $\{P_k\}_{k=1}^N$ is a polyhedral partition of a set $P$ if $\{P_k\}_{k=1}^N$ is a partition of $P$ and the sets $\overline{P}_k$ $(k = 1, ..., N)$ are polyhedra, where $\overline{P}_k$ denotes the closure of $\overline{P}_k$.*

*Once the multi-parametric problem (3.43) has been solved off-line for a polyhedral set $X_f$ of states, the explicit solution $U^*(\mathbf{q})$ of constrained finite time optimal control problem (3.42) is available as a piecewise affine function of $\mathbf{q}$, the state-feedback of the MPC (see next Chapter) law is simply*

$$\mathbf{u}(t) = [I_m \;\; 0 \;\; ... \;\; 0] \;\; U^*(\mathbf{q}(t)).$$

*In [BMDP02] it is proven that by using a multi-parametric solver the computation of MPC action becomes a simple piecewise affine function evaluation.*

## 3.7 Notes at the end of chapter

This Chapter serves as the support for the work developed. It does not contain original results. Here we studied the optimal control problem. The mathematical ideas introduced in this Chapter are developed in more depth in a number of well-written books, for example, Athans and Falb [AF66].

The problem of optimal control can then be stated as: "Determine the control functions that will cause a system to satisfy the physical constraints and, at the same time, minimize (or maximize) some performance criterion". A mathematical formulation of optimal control problems was given in Section 3.2. Then in section 3.3 we presented the necessary conditions of optimality in the form of a Maximum Principle. Such conditions are known as the Pontryagin Maximum Principle. In the search for an optimal solution, especially when the NCO are the tool of choice, to guarantee the existence of a solution is of prime importance. So, Section 3.4, based in [Cla89], gives the deductive method in optimization. In Section 3.5 we focus on the special case when the system dynamics are linear and the cost is quadratic.

# Chapter 4

# Model Predictive Control

## 4.1 Introduction

The *Model Predictive Control* (MPC), also known as *Moving-horizon control*, is an optimization-based control technique that has received an increasing research interest and has been widely applied in industry.

MPC has a success story in industry. It was initially developed and used in the 1960's by practitioners from the process engineering community, where it is still often referred to by commercial names such as Dynamic Matrix Control, Generalized Predictive Control, and others. Several thousands of applications have been reported through surveys carried out with the main software vendors [QB97]. Most of the applications reported are in the chemical processes industries, especially petrol refining processes, where the use of an optimized control strategy immediately brings about substantial economic benefits. Some world experts in control theory state that MPC is the only advanced control technique that had a significant and widespread impact on industrial process control [Mac02].

Research in MPC has been very active in the recent years (see Table 4.1). There has been an intense and increasing research effort addressing a wide range of fundamental issues such as stability, robustness, performance analysis and state estimation (see e.g. survey [MRRS00]). The application range of MPC have grown and have been included the so-called fast systems, such as robotic systems, automobile, and aeronautics applications (see [Ala06a]). These new applications are possible not only due to technological developments (increasing computational power at lower cost), but also due to theoretical developments in specific dynamic optimization algorithms [DBS05] and due to developments at a more fundamental level (e.g. enabling MPC to use discontinuous feedbacks and address nonholonomic systems). See, for example [Fon02a, FMG07].

Camacho and Bordons presented in [CB04] an interesting analogy that describes perfectly the predictive control method, this control strategy relating to driving a car. The usual way of driving a car is similar to the operation of a predictive controller:

| **Year** | **2002** | **2003** | **2004** | **2005** | **2006** | **2007** | **2008** | **2009** | **2010** |
|---|---|---|---|---|---|---|---|---|---|
| # **articles** | 320 | 359 | 337 | 464 | 495 | 491 | 518 | 524 | 559 |

Table 4.1: Research in MPC in the recent years (number of articles in ISI journals).

- The driver looks forward, he realizes the road to follow and the cars that are there;
- Knowing how the car reacts, he takes the control actions needed to follow the path correctly and safely (for example, applying the brake or accelerator and steering wheel to make the curves).

MPC can be also comparable to the strategy of a chess player: prediction of opponent moves, only a few moves ahead can be plan (not the whole game, good players think several moves ahead - long prediction horizon), an unexpected move from the opponent implies that the strategy must be change. So,

- The situation on the chess board is evaluated, i.e., the state of the process is measured and assessed;
- Future moves are considered, i.e., a mathematical algorithm calculates the optimal sequence of actions;
- The first move of the selected sequence is applied, i.e., a new set point is sent to the actuators.

This sequence is repeated after the opponent has made his move. An important advantage of such a system is that the mathematical algorithm can take limits and constraints into account when deriving the optimal control sequence.

The basic concepts pertaining to any predictive controller are the same and focus on three main points, as summarized in [CB04]:

**(i)** Using a model (linear or not) system for predicting the process output at future time instants (prediction horizon);

**(ii)** Calculating of a sequence of control inputs that minimize a given objective function;

**(iii)** Implementing the strategy of the receding horizon, where at each instant of the control prediction horizon is moved forward into the future and the first control signal is applied to the sequential system.

For this reason the predictive control is also called *Receding Horizon Predictive Control* ou RHCP.

The main idea of the MPC technique is to construct a feedback law by solving on-line a sequence of open-loop optimal control problems, each of these problems using the currently

measured state of the plant as its initial state. Similarly to optimal control, MPC has an inherent ability to deal naturally with constraints on the inputs and on the state.

Linear MPC refers to a family of MPC schemes in which linear models are used to predict the system dynamics, even though the dynamics of the closed-loop system is nonlinear due to the presence of constraints. Linear MPC approaches have found successful applications. Currently, the theory of linear MPC is well developed:

- An available option in the market;
- Best choice in certain industrial fields (petrochemical, glass, paper, etc.);
- Deep theoretical background.

Many systems are, however, in general inherently nonlinear. Traditionally, MPC schemes with guaranteed stability for nonlinear systems impose conditions on the open-loop optimal control problem that either lead to some demanding hypotheses on the system or make the on-line computation of the open-loop optimal control very hard.

Most practitioners of MPC methods know that for some systems, by an appropriate choice of some parameters of the objective function and horizon (obtained by trial-and-error and some empirical rules), it is possible to obtain stabilizing trajectories without imposing demanding artificial constraints. However, their achievements are often not supported by any theoretical result. Also, "playing" with the design parameters and test the result with simulations is an option frequently criticized by researchers (see e.g. [BGW90]).

The work of [Fon01] reduce the gap between theory and practice. He propose a very general framework of MPC for systems satisfying very mild hypotheses. The design parameters of the MPC strategy are chosen in order to satisfy a certain (sufficient) stability condition. Then, the closed-loop system resulting from applying MPC will have the desirable stability properties guaranteed.

Some nonlinear systems cannot be stabilized by a continuous feedback as was first noticed in [SS80] and [Bro83]. As seen in Chapter 2, among such systems are the nonholonomic systems, which frequently appear in practice.

Another contribution of [Fon01] was to relax a common assumption of all previous continuous-time MPC schemes: the continuity of the controls solving the open-loop optimal control problems as well as the continuity of the resulting feedback laws. This assumption, in addition to being very difficult to verify, was a major obstacle in enabling MPC to address a broader class of nonlinear systems.

More recently, in the last couple of years, an alternative MPC scheme with guaranteed stability has been developed, the so-called unconstrained MPC scheme [Gru09, GPSW10, GvLPW10].

## 4.2 Problem formulation

Consider a nonlinear plant with input constraints, where the evolution of the state after time $t_0$ is predicted by the model:

$$\left|\begin{array}{ll} \dot{\mathbf{q}}(s) = f\left(s, \mathbf{q}(s), \mathbf{u}(s)\right), & \text{a.e.} \quad s \geq t_0, \\ \mathbf{q}(t_0) = \mathbf{q}_{t_0} \in Q_0, & \\ \mathbf{q}(s) \in Q, \qquad \text{for all} \quad s \geq t_0, & \\ \mathbf{u}(s) \in U\left(s\right), \qquad \text{a.e.} \quad s \geq t_0. & \end{array}\right. \tag{4.1}$$

The data of this model comprise

- A set $Q_0 \subset \mathbb{R}^n$ containing all possible initial states at the initial time $t_0$,
- A set $Q \subset \mathbb{R}^n$ defining an admissible region of the state spaces,
- A vector $\mathbf{q}_{t_0} \in Q_0$, the state of the plant measured at time $t_0$,
- A given function $f : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$,
- A multifunction $U\left(t\right) : \mathbb{R} \rightrightarrows \mathbb{R}^m$ of possible sets of control values.

We assume this system to be asymptotically controllable on $Q_0$ and that for all $t \geq t_0$, $f\left(t, 0, 0\right) = 0$. We further assume that the function $f$ is continuous and locally Lipschitz with respect to the second argument.

The objective is to obtain a feedback law that (asymptotically) drives the state of our plant to the origin. This task is accomplished by using a MPC strategy.

Consider a sequence of sampling instants

$$\pi := \left\{t_i\right\}_{i \geq 0},$$

with a constant inter-sampling time $\delta > 0$ such that,

$$t_{i+1} = t_i + \delta, \ \forall i \geq 0.$$

Consider also the control horizon $T_c$ and prediction horizon $T_p$, with $\delta < T_c \leq T_p$, to be multiples of $\delta$ ($T_c = N_c\delta$ and $T_p = N_p\delta$ with $N_c, N_p \in \mathbb{N}$) and an auxiliary control law

$$\mathbf{k}^{aux} : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^m.$$

The feedback control is obtained by repeatedly solving online open-loop optimal control problems $P\left(t_i, \mathbf{q}_{t_i}, T_c, T_p\right)$ at each sampling instant $t_i \in \pi$, every time using the current measure of the state of the plant $\mathbf{q}_{t_i}$.

$$P\left(t,\mathbf{q}_t,T_c,T_p\right)\begin{cases}\text{Minimize} & \int_t^{t+T_p} L\left(s,\mathbf{q}(s),\mathbf{u}(s)\right)ds \quad +W\left(t+T_p,\mathbf{q}\left(t+T_p\right)\right), \\ \text{subject to:} & \dot{\mathbf{q}}(s)= f\left(s,\mathbf{q}\left(s\right),\mathbf{u}\left(s\right)\right), \quad \text{a.e. } s\in\left[t,t+T_p\right], \\ & \mathbf{q}\left(t\right)=\mathbf{q}_t, \\ & \mathbf{q}(s)\in Q, \quad \forall s\in\left[t,t+T_p\right], \\ & \mathbf{u}\left(s\right)\in U\left(s\right), \quad \text{a.e. } s\in\left[t,t+T_c\right], \\ & \mathbf{u}(s)=\mathbf{k}^{aux}\left(s,\mathbf{q}(s)\right), \quad \text{a.e. } s\in\left[t+T_c,t+T_p\right], \\ & \mathbf{q}\left(t+T_p\right)\in S.\end{cases} \tag{4.2}$$

The domain of this optimization problem is the set of admissible processes, namely pairs $(\mathbf{q},\mathbf{u})$ comprising a measurable control function $\mathbf{u}$ and the corresponding absolutely continuous state trajectory $\mathbf{q}$ which satisfy the constraints of $P\left(t,\mathbf{q}_t,T_c,T_p\right)$. The problem is said to be feasible if there exists at least an admissible process. A process $(\overline{\mathbf{q}},\overline{\mathbf{u}})$ is said to solve $P\left(t,\mathbf{q}_t,T_c,T_p\right)$ if it globally minimizes

$$\int_t^{t+T_p} L\left(s,\mathbf{q}(s),\mathbf{u}(s)\right)ds + W\left(t+T_p,\mathbf{q}\left(t+T_p\right)\right) \tag{4.3}$$

among all admissible processes.

The notation adopted here is as follows.

- The variable $t$ represents real time;
- The variable $s$ denotes the time variable used in the prediction model;
- The vector $\mathbf{q}_t$ denotes the actual state of the plant measured at time $t$.
- The process $(\mathbf{q},\mathbf{u})$ is a pair trajectory/control obtained from the model of the system;

  (The trajectory is sometimes denoted as $\mathbf{q}(s;t,\mathbf{q}_t,\mathbf{u})$ to make explicit the dependence on the initial time, initial state, and control function).
- The pair $(\overline{\mathbf{q}},\overline{\mathbf{u}})$ denotes the optimal solution to an open-loop optimal control problem;
- The process $(\mathbf{q}^*,\mathbf{u}^*)$ is the closed-loop trajectory and control resulting from the MPC strategy.

We call *design parameters* to the variables present in the open-loop optimal control problem that are not from the system model (i.e., variables that can be chosen), these comprise:

- The control horizon $T_c$ and prediction horizon $T_p$;
- A terminal cost function $W:\mathbb{R}\times\mathbb{R}^n\to\mathbb{R}$, and a running cost function $L:\mathbb{R}\times\mathbb{R}^n\times\mathbb{R}^m\to\mathbb{R}$;

- The terminal constraint set $S \subset \mathbb{R}^n$;
- The auxiliary control law $\mathbf{k}^{aux} : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^m$.

The MPC conceptual algorithm [FMG07] consists of performing the following steps at a certain instant $t_i$ (see Figure 4.1):

**MPC conceptual algorithm**

1. Measure the state of the plant $\mathbf{q}^*(t_i)$;
2. Comput the open-loop optimal control $\overline{\mathbf{u}} : [t_i, t_i + T_c] \to \mathbb{R}^n$ solution of the problem $P(t_i, \mathbf{q}^*(t_i), T_c, T_p)$;
3. Apply to the plant the control $\mathbf{u}^*(t) := \overline{\mathbf{u}}(t; t_i, \mathbf{q}^*(t_i))$ in the interval $[t_i, t_i + \delta[$. The remaining control $(\overline{\mathbf{u}}(t), t \geq t_i + \delta)$ is discarded;
4. Repeat the procedure from 1. for the next sampling instant: $t_{i+1} = t_i + \delta$, using the new measure of the state of the plant $\mathbf{q}(t_{i+1})$.

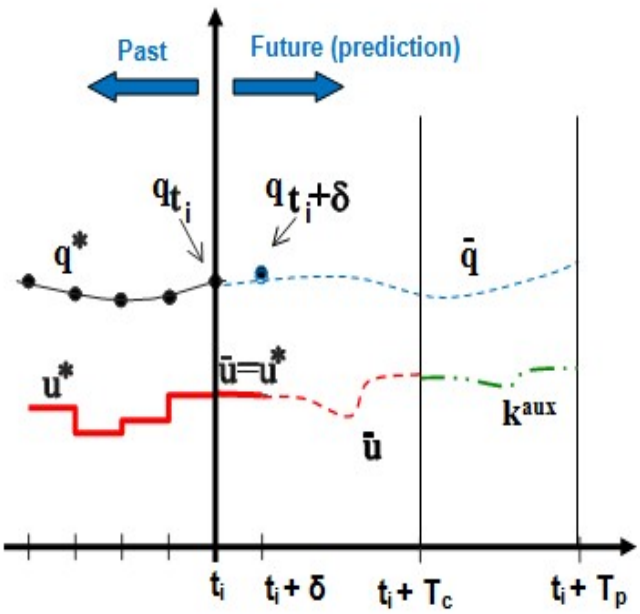


Figure 4.1: The MPC strategy (adapted from [Fon01]).

The resultant control law is a "sampling-feedback" control since during each sampling interval, the control $\mathbf{u}^*$ is dependent on the state $\mathbf{q}^*(t_i)$ the resulting trajectory is given by:

$$\begin{cases} \dot{\mathbf{q}}^*(t) = f\left(t, \mathbf{q}^*(t), \mathbf{u}^*(t)\right) & t \geq t_0, \\ \mathbf{q}^*(t_0) = \mathbf{q}_{t_0}, \end{cases}$$

where

$$\mathbf{u}^*(t) = \mathbf{k}\left(t, \mathbf{q}^*\left(\lfloor t \rfloor_\pi\right)\right) := \overline{\mathbf{u}}\left(t; \lfloor t \rfloor_\pi, \mathbf{q}^*\left(\lfloor t \rfloor_\pi\right)\right) \qquad t \geq t_0$$

and the function $t \to \lfloor t \rfloor_\pi$ gives the last sampling instant before $t$, that is:

$$\lfloor t \rfloor_\pi := \max_i \left\{t_i \in \pi : t_i \leq t\right\}.$$

This allows the MPC framework to overcome the inherent difficulty of defining solutions to differential equations with discontinuous feedbacks. In this way, the class of nonlinear systems potentially addressed by MPC is enlarged, including, for example, nonholonomic systems.

## 4.3 Stability

Two different classes of MPC schemes with guaranteed stability can be seen in the literature:

1. A scheme that uses a control Lyapunov function (CLF) or a functional in the case of infinite-dimensional systems, as additional terminal cost term, known as CLF MPC scheme;

2. A scheme that uses a controllability assumption in terms of the stage cost instead, known as Unconstrained MPC scheme.

In MPC, a infinite horizon optimal control problem is approximated by a sequence of finite horizon problems in a receding horizon fashion with the use of a finite prediction horizon, the stability is not generally guaranteed. To safeguard asymptotic stability a terminal constraints and an additional terminal cost function are often used, see Mayne et al. [MRRS00, Fon01]. On the other hand, unconstrained MPC schemes, in the sense that no terminal constraint is used, are desirable for computational reasons [Gru09, GPSW10, GvLPW10].

### 4.3.1 CLF MPC schemes

For fixed finite horizon, the closed-loop trajectory of the system ($\mathbf{q}^*$) does not necessarily coincide with the open-loop trajectory ($\overline{\mathbf{q}}$) solution to the OCP. Hence, the fact that MPC will lead to a stabilizing closed-loop system is not guaranteed a priori, and is highly dependent on the design parameters of the MPC strategy.

In [Fon01] it was guaranteed the stability of the resultant closed loop system, by choosing the design parameters to satisfy a certain stability condition. We are going to considered the following hypotheses:

**Hypotheses**

**H1** For all $t \in \mathbb{R}$ the set $U(t)$ contains the origin, and $f(t, 0, 0) = 0$.

**H2** The function $f$ is continuous, and $\mathbf{q} \longmapsto f(t, \mathbf{q}, \mathbf{u})$ is locally Lipshitz continuous for every pair $(t, \mathbf{u})$.

**H3** The set $U(t)$ is compact for all $t$, and for every pair $(t, \mathbf{q})$ the set $f(t, \mathbf{q}, U(t))$ is convex.

**H4** The function $f$ is compact on compact sets of $\mathbf{q}$, more precisely, given any compact set $Q \subset \mathbb{R}^n$, the set

$$\{\|f(t, \mathbf{q}, \mathbf{u})\| : t \in \mathbb{R}_+, \mathbf{q} \in Q, \mathbf{u} \in U(t)\}$$

is compact.

**H5** Let $Q_0 \subset \mathbb{R}^n$ be a compact set containing all possible initial states. The system 4.1 is uniformly asymptotically controllable on $Q_0$.

The use of the framework, explained in the previous section, to stabilize nonlinear systems is illustrated in the following result, for which the proof can be found in [Fon01]. It asserts that the feedback controller resulting from the application of the MPC strategy is a stabilizing controller, as long as the design parameters satisfy the stability condition **(SC)** below (see the next theorem).

**Stability conditions**

**SC** For the system 4.1 the design parameters, $T$ - the time horizon; $L$ and $W$ - the objective functions and $S$ - the terminal constraint set; satisfy:

**SC1** The set $S$ is closed and contains the origin.

**SC2** The function $L$ is continuous $L(., 0, 0) = 0$, and there is a continuous positive definite and radially unbounded function $M : \mathbb{R}^n \times \mathbb{R}_+$ such that $L(t, \mathbf{q}, \mathbf{u}) \geq M(\mathbf{q})$ for all $(t, \mathbf{u}) \in \mathbb{R}_+ \times \mathbb{R}^m$. Moreover, the "extended velocity set"

$$\{(v, l) \in \mathbb{R}^n \times \mathbb{R}_+ : v = f(t, \mathbf{q}, \mathbf{u}), \quad l \geq L(t, \mathbf{q}, \mathbf{u}), \quad \mathbf{u} \in U(t)\}$$

is convex for all $(t, \mathbf{q})$.

**SC3** The function $W$ is positive semi-definite and continuously differentiable.

**SC4** The time horizon $T$ is such that, the set $S$ is reachable in time $T$ from any initial state and from any point in the generated trajectories, i.e., there exists a set $Q$ containing $Q_0$ such that for each pair $(t_0, \mathbf{q}_0) \in \mathbb{R}_+ \times Q$ there exists a control $\overline{\mathbf{u}} : [t_0, t_0 + T_p] \to \mathbb{R}^m$ such that $\overline{\mathbf{u}}(t) \in U$ for $t \in [t_0, t_0 + T_c]$ and

$$\overline{\mathbf{u}}(t) = \mathbf{k}^{aux}(t, \mathbf{q}(t)) \quad \text{for } t \in [t_0 + T_c, t_0 + T_p]$$

satisfying

$$\mathbf{q}\left(t_0 + T_{p'}; t_0, \mathbf{q}_0, \overline{\mathbf{u}}\right) \in S.$$

Also, for all control functions $\overline{\mathbf{u}}$ in the conditions above

$$\mathbf{q}\left(t; t_0, \mathbf{q}_0, \overline{\mathbf{u}}\right) \in Q \text{ for all } t \in \left[t_0, t_0 + T_p\right].$$

**SC5** There exists a scalar $\varepsilon > 0$ such that for every time $t \in [T_p, \infty[$ and each $\mathbf{q}_t \in S$, we can choose a control function $\mathbf{k}^{aux}$ continuous from the right at $t$ satisfying

**SC5a** $\quad W_t\left(t, \mathbf{q}_t\right) + W_{\mathbf{q}}\left(t, \mathbf{q}_t\right) \cdot f\left(t, \mathbf{q}_t, \mathbf{k}^{aux}\left(t\right)\right) \leq -L\left(t, \mathbf{q}_t, \mathbf{k}^{aux}\left(t\right)\right)$

and

**SC5b** $\quad \mathbf{q}\left(t + r; t, \mathbf{q}_t, \mathbf{k}^{aux}\right) \in S$ for all $r \in [0, \varepsilon]$.

The set of conditions in **(SC)** can be seen as divided into two types: the first type consists of the conditions guaranteeing the existence of solutions to the OCPs; the other type comprises the conditions ensuring that the closed-loop trajectory is actually driven towards the origin. The condition **(SC5a)** has similarities with the decrescence condition satisfied by Control Lyapunov Functions (CLF), the main difference being that it is merely required to be satisfied within the terminal set S. The condition **(SC5b)** simply states that the terminal set S is invariant under the control chosen in **(SC5a)**. So, the result on stability is:

**Theorem 18** *[Fon99] Assume that the system satisfies hypotheses **H1**-**H5**. If the design parameters are choose to satisfy **SC** then the cloosed loop system resulting from the application of the MPC strategy is asymptotically stable.*

Another result from [Fon99] says that it is always possible to choose the design parameters satisfying **SC**.

**Theorem 19** *[Fon99] Assume **H1**-**H5**. Then it is always possible to find design parameters $S, T, L,$ and $W$ to satisfy **SC**.*

The proof of these results is supplied in [Fon99].

The result has the attractive consequence that it is always possible to design a stabilizing MPC strategy using free terminal state optimal control problems.

As an immediate consequence of these two results [Fon99], we are able to design a stabilizing feedback for any nonlinear system belonging to the large class of systems that satisfies the assumptions **H1**-**H5**, using the MPC startegy.

In [Fon00] it was shown how most of the MPC schemes covered by the general framework described above, and emphasize the prominent role of the stability conditions, mainly (**SC5**), in ensuring the stabilizing properties of each approach (e.g., terminal state constrained to the origin, infinite horizon, dual-mode approach, terminal-cost based approach, ...).

### 4.3.2 Unconstrained MPC schemes

The results for discrete-time systems from Grüne [Gru09], Grüne et al. [GPSW10] and [GvLPW10] use an MPC scheme with neither terminal cost functions nor terminal constraints. A controllability assumption allows to guarantee stability and to estimate the performance compared to an infinite horizon optimal controller.

In the paper of [RA11] a continuous-time version of recent results on unconstrained nonlinear MPC schemes was presented. Reble and Allgöwer extended the results on Model Predictive Control without terminal constraints and terminal cost functions from the discrete-time case to continuous-time.

Consider the problem:

$$\begin{array}{ll} \text{Minimize} & J_\infty(\mathbf{q}_0) = \int_0^\infty L(\mathbf{q}(t), \mathbf{u}(t))\, dt, \\ \text{subject to:} & \dot{\mathbf{q}}(t) = f(\mathbf{q}(t), \mathbf{u}(t)), \\ & \mathbf{q}(0) = \mathbf{q}_0, \end{array} \tag{4.4}$$

where $\mathbf{q}(t) \in \mathbb{R}^n$ is the state vector and $\mathbf{u}(t) \in U \subset \mathbb{R}^m$ is the control input. Assume that the function $f$ is continuously differentiable and the constraint set $U$ is compact, convex, and contains the origin in its interior. Assume also that $f(0,0) = 0$. The cost $L : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^+$ is continuous in both arguments. The optimal cost of this infinite horizon problem is denoted by $J^*_\infty(\mathbf{q}_0)$.

In [RA11] it was guaranteed the stability of the resultant closed loop system. We are going to considered the following assumption:

**CA** [RA11] **(Controllability Assumption)** For all $\mathbf{q}_0$, there exists a piecewise continuous input trajectory $\widehat{\mathbf{u}}(.;\mathbf{q}_0)$ with $\widehat{\mathbf{u}}(t;\mathbf{q}_0) \in U$ for all $t \geq 0$ and with corresponding state trajectory $\widehat{\mathbf{q}}$ such that

$$L(\widehat{\mathbf{q}}\ (t); \widehat{\mathbf{u}}(t;\mathbf{q}_0)) \geq \beta(t) \min_{\mathbf{u}\in U} L(\mathbf{q}_0, \mathbf{u}), \quad \forall t \in \mathbb{R}^+$$

for some function $\beta : \mathbb{R}^+ \to \mathbb{R}^+$ that is a continuous, positive, and absolutely integrable function with

$$\lim_{t\to\infty} \beta(t) = 0.$$

Define,

$$\overline{J}_0 = \min \int_{t_i}^{t_i+T} L(\mathbf{q}(t), \mathbf{u}(t))\, dt.$$

Solving this optimal control problem we have the solution $(\overline{\mathbf{q}}, \overline{\mathbf{u}})$ in the interval $[t_i, t_i + T]$, where $\mathbf{q}(t_i) = \mathbf{q}_{t_i}$ is given.

Define also,

$$\overline{J}_{\delta} = \min \int_{t_i+\delta}^{t_i+T+\delta} L\left(\mathbf{q}(t), \mathbf{u}(t)\right) dt.$$

Solving this optimal control problem we have the solution $\left(\overline{\overline{\mathbf{q}}},\overline{\overline{\mathbf{u}}}\right)$ (that could be diferent from $(\overline{\mathbf{q}},\overline{\mathbf{u}})$) in the interval $[t_i+\delta, t_i+\delta+T]$, where $\overline{\overline{\mathbf{q}}}\left(t_i+\delta\right) = \overline{\mathbf{q}}\left(t_i+\delta\right)$.

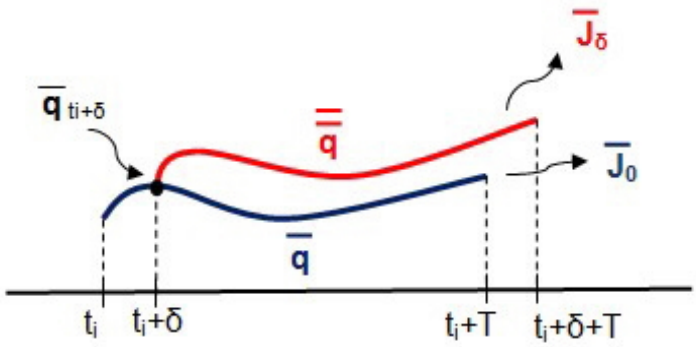


Figure 4.2: Different optimal control problems and horizons considered.

Define $J_{\infty}^{MPC}\left(\mathbf{q}_0\right)$ as is the infinite horizon cost resulting from the application of the MPC. So, the result from [RA11] regarding stability and suboptimality for the unconstrained MPC scheme is in the following theorem on stability:

**Theorem 20** *[RA11]**(Stability and Suboptimality)** For sampling time $\delta$, prediction horizon $T$ and Controllability Assumption **CA**, the following holds*

**i)** $\exists \alpha \in (0,1]$ *such that*

$$\overline{J}_{\delta} - \overline{J}_0 \leq -\alpha \int_{t_i}^{t_i+T} L\left(\mathbf{q}(t), \mathbf{u}(t)\right) dt;$$

**ii)** *For all* $\mathbf{q}_0 \in \mathbb{R}^n$.

$$\alpha J_{\infty}^{MPC}\left(\mathbf{q}_0\right) \leq J_{\infty}^{*}\left(\mathbf{q}_0\right);$$

**iii)** *Asymptotic stability of the closed loop system is guaranteed.*

The suboptimality index $\alpha$ can be estimated by

$$\alpha = 1 + \left(\overline{\beta} - 1\right) \frac{B\left(T\right)}{\overline{\beta}\delta},$$

where $\overline{\beta}$ can be constructed by solving a linear program as in Theorem 4 in [RA11].

## 4.4 Notes at the end of Chapter

The Model Predictive Control (MPC) technique constructs a feedback law by solving on-line a sequence of open-loop optimal control problems, each of them using the currently measured state of the plant as its initial state. Similarly to optimal control, MPC has an inherent ability to deal naturally with constraints on the inputs and on the state. Since the controls are obtained by optimizing some criterion, the method possesses some desirable performance properties, and also intrinsic robustness properties [MS97].

There are two control schemes where stability is assured: CLF MPC and Unconstrained MPC schemes. The first class of schemes use a control Lyapunov function and in [Fon01] it was guaranteed the stability of the resultant closed loop system, by choosing the design parameters to satisfy a certain stability condition. The framework proposed in [Fon03] that describes how a continuous-time MPC framework using a positive inter-sampling time, combined with the use of an appropriate concept of solution to a differential equation, can address nonholonomic systems. The second class of schemes uses a controllability assumption in terms of the stage cost instead. The framework of [RA11] extended the results on Model Predictive Control without terminal constraints and terminal cost functions from the discrete-time case, from Grüne [Gru09], Grüne et al. [GPSW10] and [GvLPW10], to continuous-time.

This Chapter serves as the support for the work developed. It does not contain original results.

# Chapter 5

# Optimal Control and MPC algorithms

## 5.1 Introduction

In optimal Control we first have to distinguish whether we are looking into an open-loop control solution or a closed-loop control solution.

If we are looking into an open-loop solution we may apply (i) undirect methods or (ii) direct methods (see Figure 5.1).

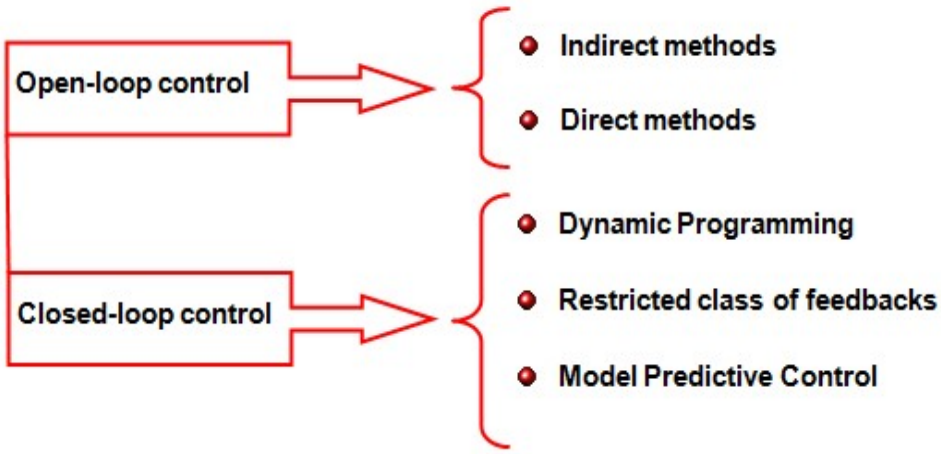


Figure 5.1: Open-loop control solutions and closed-loop control solutions.

The undirect methods are based on applying the NCO in the form of Maximum Principle and then solving the two-point boundary value problem that results from it. Further details are given in Section 3.3.1. These methods are known to suffer from robustness problems [Bet01] and, since we are dealing with highly nonlinear system with constraints, we have chosen not to develop algorithms based on these methods.

The direct methods are based on transforming the OCP into a nonlinear programming problem (NLP). The control and state function are approximated by a finite number of parameters (polynomial approximation, piecewise constant approximation, etc.) and these parameters are treated as optimization variables in the NLP.

The direct methods have been prefered in the last years. Although the problem dimension might be larger, it is usually sparse and there are solvers adequately built for treating these problems.

Most recent Optimal Control packages are based on directed methods:

· RIOTS [Sch98] - Matlab2 toolbox for solving optimal control problems.RIOTS_95 is a group of programs and utilities, written mostly in C and designed as a toolbox for Matlab, that provides an interactive environment for solving a very broad class of optimal control problems;

· DIDO [Ros07] - is a MATLAB program for solving hybrid optimal control problems;

· GPOPS [RDG$^+$] - GPOPS (which stands for "General Pseudospectral OPtimal Control Software") is an open-source MATLAB optimal control software that implements the Gauss and Radau hp-adaptive pseudospectral methods. These methods approximate the state using a basis of Lagrange polynomials and collocate the dynamics at the Legendre-Gauss-Radau points. These methods share the property that they can be written equivalently in either in differential form or in implicit integral form;

· PROPT [RE10] - This software package is intended to help solve dynamic optimization problems. The goal of PROPT is to make it possible to input such problem descriptions as simply as possible, without having to worry about the mathematics of the actual solver. Once a problem has been properly modeled, PROPT will take care of all the steps necessary in order to return a solution;

· ACADO [HF11] - ACADO Toolkit is a software environment and algorithm collection for automatic control and dynamic optimization. It provides a general framework for using a great variety of algorithms for direct optimal control, including model predictive control, state and parameter estimation and robust optimization.

We will develop in Section 5.3.1 and 5.3.2 the implementation of direct methods when the control functions are approximated by piecewise constant function and polynomials, respectively.

Regarding closed-loop optimal control, we have seen in Chapter 3 that solutions can be easily found for the Linear Quadratic Regulator problem. For general problems a feedback law might be obtained by (i) Dynamic Programming / Hamilton Jacobi Equation, (ii) by restricting the class of feedback functions (e.g., affine, bang-bang) and thereby obtain an approximate solution, or (iii) by using MPC, that generates a feedback by solving a sequence of open-loop OCPs and thereby also generates an aproximate solution.

Only (i) Dynamic Programming would give us the optimal closed-loop control, but the computational cost is very high. We discuss (ii) restricted class of feedback function in sections 5.3.3 and 5.3.4 and (iii) MPC in chapters 4, 6 and 7.

## 5.2 The test problem

In this Chapter, we report results on the implementation of an Optimal Control Algorithm and an stable MPC strategy for a wheeled robot, where finite parameterizations for discontinuous control functions are used, resulting in efficient computation of the control strategies.

We will explore a simple case of a differential-drive mobile robot moving on a plane (see Figure 2.4), and represented by the following kinematic model:

$$\begin{cases} \dot{x} &= v\cos\theta, \\ \dot{y} &= v\sin\theta, \\ \dot{\theta} &= w, \end{cases} \tag{5.1}$$

with the state $\mathbf{q} = (x, y, \theta)$ and where $(x, y)$ is the position in the plane of the midpoint of the axle connecting the rear wheels and $\theta$ denotes the heading angle measured anticlockwise from the $x$-axis. The velocity control of the two rear wheels determines the translation velocity of the robot $v$ and the angular velocity $w$.

Considere an auxiliary feedback control function $\mathbf{k}^{aux}$ defined as:

$$\begin{array}{ll} \text{if } \|(x,y)\| < 0.05 & \text{then} \\ & v = 0 \text{ and } w = -\frac{\pi}{4} \times \theta, \\ & \text{else} \\ & v = 0.1 \text{ and } w = -\frac{\pi}{4} \times [\theta - \phi(x,y)]\,. \end{array} \tag{5.2}$$

where

$$\phi(x,y) = \begin{cases} 0 & \text{se} \quad \|(x,y)\| \leq \varepsilon, \\ -(\pi/2)\,\text{sign}\,(y) & \text{se} \quad \|(x,y)\| > \varepsilon, x = 0, y \neq 0, \\ \tan^{-1}(y/x) + \pi & \text{se} \quad \|(x,y)\| > \varepsilon, x > 0, \\ \tan^{-1}(y/x) & \text{se} \quad \|(x,y)\| > \varepsilon, x < 0. \end{cases}$$

This function is easily implementable in Matlab (which have a four-quadrant inverse tangent built-in function) as $\phi(x, y) = ATAN2(-y, -x)$.

This stabilizing feedback will prove, for any initial condition $\mathbf{q}_0$, a good initial guess for the open-loop control.

As we will see in Chapter 6 (section 6.2), the guarantee of stability of the resulting closed loop system can be given by a choice of design parameters satisfying a sufficient stability condition. The following set of design parameters guarantees stability:

**Terminal set $S$:**

Define the terminal set $S$ to be the set of states heading towards the origin of the plane together with the origin of the plane, that is:

$$S = \left\{ \mathbf{q} = (x, y, \theta) \in \mathbb{R}^2 \times [-\pi, \pi] : (x, y, \theta) \in \Theta \vee (x, y) = (0, 0) \right\}, \tag{5.3}$$

where the target set is

$$\Theta = \left\{ \mathbf{q} = (x, y, \theta) \in \mathbb{R}^2 \times [-\pi, \pi] : \|(x, y)\| \leq 0.05, |\theta| \leq 0.1 \right\}.$$

**The time horizon $T$:**

The set $S$ can be reached from any state if we define the horizon to be the time to complete an 180 degrees turn, that is

$$T = \pi / w_{\max} = \pi. \tag{5.4}$$

**The functions $L$ and $W$:**

Consider the functions $L(\mathbf{q})$ and $W(\mathbf{q})$ used in [FM03a]

$$L(\mathbf{q}) = x^2 + y^2 + \theta^2 \tag{5.5}$$

and

$$W(\mathbf{q}) = \frac{1}{3}\left(r^3 + |\theta|^3\right) + r\theta^2, \quad \text{with} \quad r = \sqrt{x^2 + y^2}. \tag{5.6}$$

The Optimal Control Problem to be solved is

$$\begin{array}{lll} \text{Minimize} & \int_0^T L(t, \mathbf{q}(t), \mathbf{u}(t))dt + W(\mathbf{q}(T)), & \\ \text{subject to} & \dot{\mathbf{q}}(t) = f(t, \mathbf{q}(t), \mathbf{u}(t)), & \text{a.e. } t \in [0, T], \\ & \mathbf{q}(0) = \mathbf{q}_0, & \\ & \mathbf{q}(T) \in S \subset \mathbb{R}^n, & \\ & \mathbf{u}(t) \in U \subset \mathbb{R}^m, & \text{a.e. } t \in [0, T]. \end{array}$$

## 5.3 Optimal Control Algorithms

This section illustrates the implementation of the direct method when the control functions are approximated by piecewise constant function and polynomials respectively. We will obtain a closed-loop optimal control feedback law by restricted the class of feedback functions (polynomial closed-loop control feedbacks and bang-bang feedbacks).

### 5.3.1 Piecewise constant open-loop controls

Suppose the problem is stated like:

$$\begin{array}{ll} \text{Minimize} & \int_0^T L\left(t,\mathbf{q}\left(t\right),\mathbf{u}\left(t\right)\right)dt + W\left(\mathbf{q}\left(T\right)\right), \\ \text{subject to:} & \dot{\mathbf{q}}\left(t\right) = f\left(t,\mathbf{q}\left(t\right),\mathbf{u}\left(t\right)\right), \qquad a.e.\ t\in\left[0,T\right], \\ & \mathbf{q}\left(0\right) = \mathbf{q}_0, \\ & \mathbf{u}\left(t\right)\in U, \qquad a.e.\ t\in\left[0,T\right], \end{array} \tag{5.7}$$

that is, the problem is in the Bolza form.

By introducing an additional state variable, say $\mathbf{z}$, and the differential equation:

$$\dot{\mathbf{z}} = L\left(t,\mathbf{q}\left(t\right),\mathbf{u}\left(t\right)\right),$$

with initial condition $\mathbf{z}\left(0\right)=0$ it is possible to replace the original objective function of problem (5.7) by

$$\mathbf{z}\left(T\right) + W\left(\mathbf{q}\left(T\right)\right)$$

so, the problem now is:

$$\begin{array}{ll} \text{Minimize} & \mathbf{z}\left(T\right) + W\left(\mathbf{q}\left(T\right)\right), \\ \text{subject to:} & \dot{\mathbf{q}}\left(t\right) = f\left(t,\mathbf{q}\left(t\right),\mathbf{u}\left(t\right)\right), \qquad a.e.\ t\in\left[0,T\right] \\ & \mathbf{q}\left(0\right) = \mathbf{q}_0, \\ & \dot{\mathbf{z}} = L\left(t,\mathbf{q}\left(t\right),\mathbf{u}\left(t\right)\right), \\ & \mathbf{z}\left(0\right) = 0, \\ & \mathbf{u}\left(t\right)\in U, \qquad a.e.\ t\in\left[0,T\right], \end{array}$$

that is, the problem is in the Mayer form.

Consider a partition of the time interval:

$$\pi = \left\{t_0, t_1, ..., t_N\right\} \subset \left[0,T\right].$$

Let $U\subset\mathbb{R}^m$ be a set of possible control values ($\mathbf{u}\left(t\right)\in U$). The class of admissible open-loop control functions is

$$\mho := \left\{\mathbf{u}:\left[0,T\right]\to\mathbb{R}^m : \mathbf{u}\left(t\right) = \mathbf{u}_k,\quad \mathbf{u}_k\in U,\ t\in\left[t_{k-1},t_k\right[,\ k=1,2,...N\right\}.$$

So, we say that $(\overline{\mathbf{q}},\overline{\mathbf{u}})$ is a *minimizer for the optimal control problem*, if it minimizes the cost over all admissible arcs $\mathbf{q}$ and function $\mathbf{u}\in\mho$, that is

$$J\left(\overline{\mathbf{q}},\overline{\mathbf{u}}\right) \le J\left(\mathbf{q},\mathbf{u}\right),$$

for all absolutly continuous functions $\mathbf{u}\in\mho$, such that

$$\left\|\mathbf{q}-\overline{\mathbf{q}}\right\| < \delta,$$

for some $\delta > 0$.

The optimal control problem can be converted into a finite-dimensional optimization problem in $\mathbb{R}^{m\times N}$:

$$\min_{\mathbf{u}_0, \mathbf{u}_1, ..., \mathbf{u}_{N-1} \in \mathbb{R}^m} \quad \overline{J}\left(\mathbf{q}, \mathbf{u}_0, \mathbf{u}_1, ..., \mathbf{u}_{N-1}\right)$$

subject to the problem constraints.

The problem stated in (5.7) is in continuous time and we want to discretize it as follows:

1. For the dinamics we use the Euler's method or the Classical Runge-Kutta method.
2. For the objective function:
   (a) it can restated as a Mayer form, then it is direct, but for some problems it might not be the best option (see [Bet01]).
   (b) if it is in the Bolza form, then the integral term can be approximated by a trapezoidal discretization.

So, in the Bolza form using Euler's method we have the problem:

$$\begin{array}{ll} \text{Minimize} & \sum_{k=0}^{N-1} \frac{h}{2}\left[L\left(\mathbf{q}_{k+1}, \mathbf{u}_{k+1}\right) - L\left(\mathbf{q}_k, \mathbf{u}_k\right)\right] + W\left(\mathbf{q}_N\right), \\ \text{subject to:} & \mathbf{q}_{k+1} - \mathbf{q}_k - hf\left(\mathbf{q}_k, \mathbf{u}_k\right) = 0, \\ & \mathbf{q}_0 = \mathbf{q}_0, \end{array}$$

where $h = \frac{T}{N}$.

If a Runge-Kutta method was to be used then the dynamic equation would be:

$$\mathbf{q}_{k+1} - \mathbf{q}_k - \frac{1}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right) = 0,$$

where,

$$\begin{array}{l} k_1 = hf\left(\mathbf{q}_k, \mathbf{u}_k\right), \\ k_2 = hf\left(\mathbf{q}_k + \frac{1}{2}k_1, t_k + \frac{h}{2}\right), \\ k_3 = hf\left(\mathbf{q}_k + \frac{1}{2}k_2, t_k + \frac{h}{2}\right), \\ k_4 = hf\left(\mathbf{q}_k + k_3, t_{k+1}\right). \end{array}$$

As an example, we are going to consider the model (5.1), so we have the problem:

$$\begin{array}{ll} \text{Minimize} & \int_{t_i}^{t_i+T}\left(x^2 + y^2 + \theta^2\right) dt + \frac{1}{3}\left[\left(x^2+y^2\right)^{\frac{3}{2}} + |\theta|^3\right] + \theta^2\sqrt{x^2+y^2}, \\ \text{subject to:} & \dot{x} = v\cos\theta, \\ & \dot{y} = v\sin\theta, \\ & \dot{\theta} = w, \\ & \left(x\left(0\right), y\left(0\right), \theta\left(0\right)\right) = \left(0, 1, \frac{\pi}{2}\right). \end{array} \tag{5.8}$$

We are going to use the Mayer formulation instead of the Bolza formulation. These two formulations are equivalent, but the Mayer form yields simpler expressions, that makes it easier to code for numerical solutions. The problem is then staded as:

$$\begin{array}{ll}
\text{Minimize} & \frac{1}{3}\left[(x^2+y^2)^{\frac{3}{2}}+|\theta|^3\right]+\theta^2\sqrt{x^2+y^2}+z, \\
\text{subject to:} & \dot{x}=v\cos\theta, \\
& \dot{y}=v\sin\theta, \\
& \dot{\theta}=w, \\
& \dot{z}=x^2+y^2+\theta^2, \\
& (x(0),y(0),\theta(0),z(0))=\left(0,1,\frac{\pi}{2},0\right).
\end{array}$$

Discretizing it, we have:

$$\begin{array}{ll}
\text{Minimize} & \frac{1}{3}\left[(x^2+y^2)^{\frac{3}{2}}+|\theta|^3\right]+\theta^2\sqrt{x^2+y^2}+z, \\
\text{subject to:} & x_{k+1}=x_k+h\ v\cos(\theta_k), \\
& y_{k+1}=y_k+h\ v\sin(\theta_k), \\
& \theta_{k+1}=\theta_k+h\ w, \\
& z_{k+1}=z_k+h\left[(x_k)^2+(y_k)^2+(\theta_k)^2\right], \\
& (x(0),y(0),\theta(0),z(0))=\left(0,1,\frac{\pi}{2},0\right).
\end{array} \tag{5.9}$$

Consider that initial position of the robot is $(x_0,y_0,\theta_0)=\left(0,1,\frac{\pi}{2}\right)$, $z_0=0$, $t\in[0,20]$ and $N=100$, so $h=0.2$.

This is a nonlinear programming problem. Now we have two options of which variables we choose to be the decision variables:

**A** $[v,w]$ for $k=0,...,N-1$ are the only $2N$ decision variables and $x,y,\theta,z$ for all $k=1,...,N-1$ are computed knowing $[v,w]$ and$(x_0,y_0,\theta_0,z_0)$ by equations:

$$\left|\begin{array}{l}
x_{k+1}=x_k+h\ v\cos(\theta_k), \\
y_{k+1}=y_k+h\ v\sin(\theta_k), \\
\theta_{k+1}=\theta_k+h\ w, \\
z_{k+1}=z_k+h\left[(x_k)^2+(y_k)^2+(\theta_k)^2\right].
\end{array}\right.$$

**B** we consider $[v,w,x,y,\theta,z]$ to be the decision variables. In this case the problem dimension is larger (we have $6N$ variables) but the optimizer, in general, behave better [Bet01].

In this case the problem would have $4N$ additional equality constraints:

$$\left|\begin{array}{l}
x_{k+1}-x_k-h\ v\cos(\theta_k)=0, \\
y_{k+1}-y_k-h\ v\sin(\theta_k)=0, \\
\theta_{k+1}-\theta_k-h\ w=0, \\
z_{k+1}-z_k-h\left[(x_k)^2+(y_k)^2+(\theta_k)^2\right]=0.
\end{array}\right. \qquad k=0,...,N-1,$$

We have implemented this last option.

**Algorithm**

1. Find an initial open-loop control based on the feedback strategy (5.10):

$$\begin{array}{lll} \text{For } k=0,...N & \text{if} & \|(x,y)\| \geq 0.05, \\ & & v_k = 0.1, \\ & & w_k = -\frac{\pi}{4}\cdot\left[\theta - \phi(x,y)\right], \\ & \text{else} & \\ & & v_k = 0, \\ & & w_k = -\frac{\pi}{4}\cdot\theta. \end{array} \tag{5.10}$$

2. Solve the nonlinear programming problem with respect to $[v, w, x, y, \theta, z]$

$$\begin{array}{lll} \underset{\text{over } [v,w,x,y,\theta,z]\in\mathbb{R}^{6N}}{\text{Minimize}} & \frac{1}{3}\left[(x^2+y^2)^{\frac{3}{2}} + |\theta|^3\right] + \theta^2\sqrt{x^2+y^2} + z, & \\ \text{subject to:} & x_1 - x_0 = 0, & \\ & y_1 - y_0 = 0, & \\ & \theta_1 - \theta_0 = 0, & \\ & z_1 - z_0 = 0, & \\ & x_{k+1} - x_k - h\ v_k\cos(\theta_k) = 0, & k = 0,...N-1, \\ & y_{k+1} - y_k - h\ v_k\sin(\theta_k) = 0, & k = 0,...N-1, \\ & \theta_{k+1} - \theta_k - h\ w_k = 0, & k = 0,...N-1, \\ & z_{k+1} - z_k - h\left[(x_k)^2 + (y_k)^2 + (\theta_k)^2\right] = 0, & k = 0,...N-1. \end{array} \tag{5.11}$$

The nonlinear programming problem is solved using available packages. In this case, we used the fmincon function of Matlab from the optimization package for solving nonlinear optimization problems.

In conclusion, piecewise constant controls converge easily for this problem, but are computationally heavier and, cannot be applied to fast systems. In our simulation it took an average 186 seconds. Figure (5.2) shows our simulation result.

### 5.3.2 Polynomial open-loop control function

Here we consider that the open-loop controls are parametrized by

$$\boldsymbol{\alpha} = [\alpha_0, \alpha_1, ..., \alpha_M], \qquad \alpha_i \in \mathbb{R}^m,$$

as $M$ degree polynomial:

$$\mathbf{u}(t) = \sum_{k=0}^{M} \alpha_k\ t^k, \qquad t \in [0, T].$$

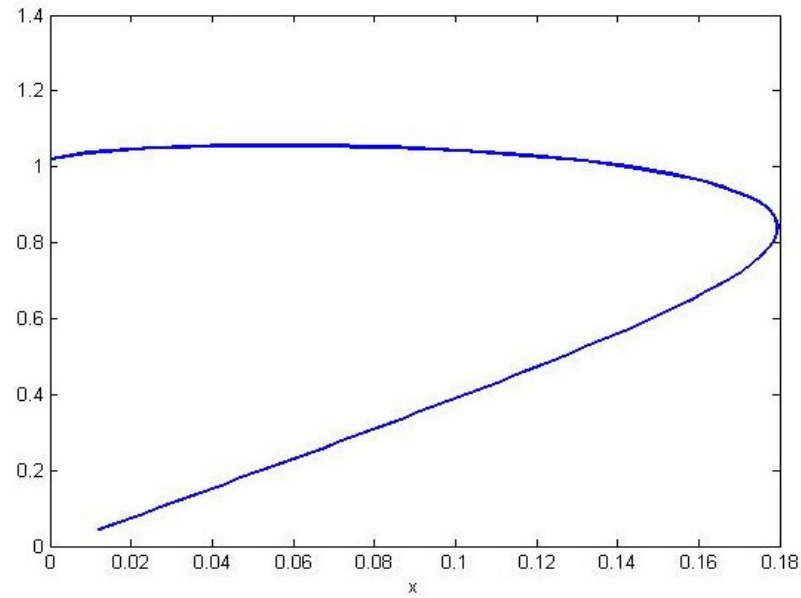


Figure 5.2: Trajectory in the plan using piecewise constant controls.

So, our class of admissible controls is

$$\mho = \left\{ \mathbf{u} : [0,T] \to \mathbb{R}^m : \quad u(t) = \sum_{k=0}^{M} \alpha_k \, t^k \quad \text{with} \quad \alpha_0, \alpha_1, ..., \alpha_M \in \mathbb{R} \right\}.$$

Discontinuous control laws cannot be adequately parametrized by these polynomials.

As an example, we are going to consider again model (5.1), which has been discretezed in the Mayer form (5.9). In this simulation we use polynomials of degree three.

**Algorithm**

1. (a) Find an initial open-loop control based on the feedback strategy (5.10):

$$\begin{array}{lll} \text{For } k = 0, ...N & \text{if} & \|(x,y)\| \geq 0.05, \\ & & v_k = 0.1, \\ & & w_k = -\frac{\pi}{4} \cdot [\theta - \phi(x,y)], \\ & \text{else} & \\ & & v_k = 0, \\ & & w_k = -\frac{\pi}{4} \cdot \theta, \end{array} \tag{5.12}$$

   (b) finds the coefficients $\boldsymbol{\alpha}\mathbf{v}$ and $\boldsymbol{\alpha}\mathbf{w}$ of the polynomials of degree 3 that fit in the $v$ and $w$ data, respectively, by minimizing the sum of the squares of the deviations of the data from the model (least-squares fit).

Note: in the end we have the vector $\boldsymbol{\alpha}$, that has 8 components (four coefficients of each polynomial):

$$\boldsymbol{\alpha} = [\boldsymbol{\alpha}\mathbf{v}, \boldsymbol{\alpha}\mathbf{w}] = [\alpha_{v_0}, \alpha_{v_1}, \alpha_{v_2}, \alpha_{v_3}, \alpha_{w_0}, \alpha_{w_1}, \alpha_{w_2}, \alpha_{w_3}].$$

2. Solve the nonlinear programming problem.

$$\begin{array}{lll}
\underset{\text{over } \boldsymbol{\alpha}\in\mathbb{R}^8}{\text{Minimize}} & \frac{1}{3}\left[(x^2+y^2)^{\frac{3}{2}} + |\theta|^3\right] + \theta^2\sqrt{x^2+y^2} + z, & \\
\text{subject to:} & x_1 - x_0 = 0, & \\
& y_1 - y_0 = 0, & \\
& \theta_1 - \theta_0 = 0, & \\
& z_1 - z_0 = 0, & \\
& x_{k+1} - x_k - h\ v_k \cos(\theta_k) = 0, & k = 0, ...N-1, \\
& y_{k+1} - y_k - h\ v_k \sin(\theta_k) = 0, & k = 0, ...N-1, \\
& \theta_{k+1} - \theta_k - h\ w_k = 0, & k = 0, ...N-1, \\
& z_{k+1} - z_k - h\left[(x_k)^2 + (y_k)^2 + (\theta_k)^2\right] = 0, & k = 0, ...N-1, \\
& v(k) = \sum_{i=0}^{3} \alpha_{v_i}\ (kh)^i, & k = 0, ...N-1, \\
& w(k) = \sum_{i=0}^{3} \alpha_{w_i}\ (kh)^i, & k = 0, ...N-1.
\end{array} \tag{5.13}$$

In this case, we used again fmincon function of Matlab from the optimization package for solving the nonlinear optimization problem.

In conclusion, polynomial controls are very fast but we had convergence problems with our example. Moreover, discontinuous controls cannot be adequately aproximated by such polynomials. Figure (5.3) shows our simulation results.

### 5.3.3 Polynomial closed-loop control feedbacks

Here we consider just the case where the control is an affine feedback:

$$\mathbf{u}(t) = \mathbf{K}\mathbf{q}(t) + \mathbf{F}, \qquad t \in [0, T],$$

where $\mathbf{K} \in \mathbb{R}^{m\times n}$ and $\mathbf{F} \in \mathbb{R}^m$. So, the optimization problem is

$$\begin{array}{ll}
\underset{\mathbf{K}\in\mathbb{R}^{m\times n},\ \mathbf{F}\in\mathbb{R}^m}{\text{Minimize}} & \int_0^T L(t, \mathbf{q}(t), \mathbf{K}\mathbf{q}(t) + \mathbf{F})dt + W(\mathbf{q}(T)), \\
\text{subject to} & \mathbf{q}_{k+1} - \mathbf{q}_k - hf(t, \mathbf{q}_k, \mathbf{K}\mathbf{q}_k + \mathbf{F}) = 0, \\
& \mathbf{q}(0) = \mathbf{q}_0, \\
& \mathbf{q}(t) \in S \subset \mathbb{R}^n, \\
& \mathbf{K}\mathbf{q}(t) + \mathbf{F} \in \mathbf{U} \subset \mathbb{R}^m.
\end{array}$$

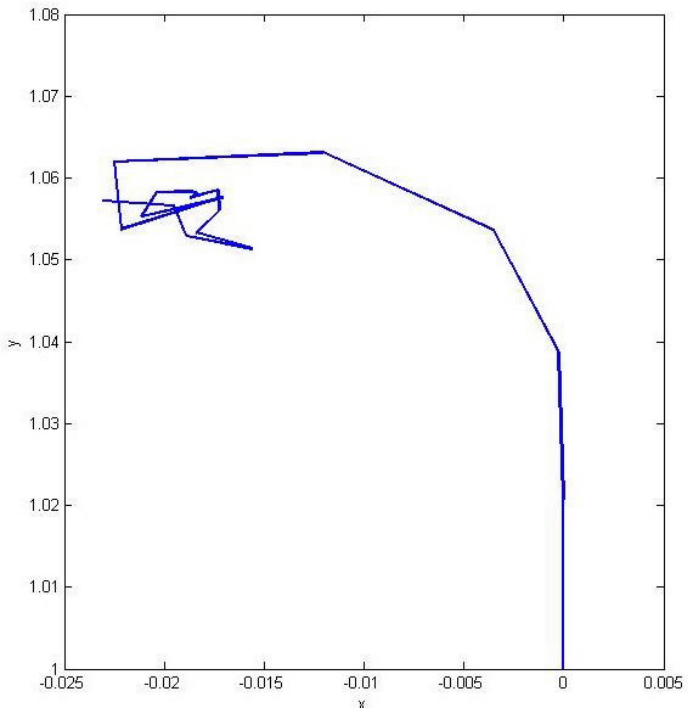


Figure 5.3: Trajectory in the plan using polynomial controls.

Since our problem is nonlinear the Optimal Control Problem might be significantly different from the affine law obtained here. We have just presented this parametrization for completeness, althought we have not implemented this problem. This might be a possible method to obtained initial guess.

### 5.3.4 Bang-Bang feedbacks

We describe here the use of discontinuous feedback control strategies of the bang-bang type, which can be described by a small number of parameters and so make the problem computationally tractable. In bang-bang feedback strategies, the control values of the strategy are only allowed to be at one of the extremes of its range. Many control problems of interest admit a bang-bang stabilizing control. Fontes and Magni ([FM03a]) describe the application of this parameterization to a unicycle mobile robot subject to bounded disturbances.

The controller used here follows closely the results in [FM03a]. We use bang-bang feedback controls. That is, for each state, the corresponding control must be at one of the extreme values of its range. The exception is the target set $\Theta$ (here a small ball centered at the origin) where the control is chosen to be zero.

The feedbacks, when outside the target set, are defined by a switching surface $\sigma(\mathbf{q}) = 0$. The control will attain its maximum or minimum value depending on which side of the surface the state is. More precisely, for each component $j = 1, \ldots, m$ of the control vector, we define

$$k_j(\mathbf{q}) = \begin{cases} 0 & if \quad \mathbf{q} \in \Theta, \\ u_j^{\max} & if \quad \sigma_j(\mathbf{q}) \geq 0, \\ u_j^{\min} & if \quad \sigma_j(\mathbf{q}) < 0. \end{cases} \tag{5.14}$$

The function $\sigma_j$ is a component of the vector function $\sigma$, and is associated with the switching surface $\sigma_j(\mathbf{q}) = 0$ which divides the state-space in two.

Since these surfaces must be parameterized in some way to be chosen in an optimization problem, we will define them to have a fixed part $\sigma^{aux}$, possibly nonlinear, and a variable part $\sigma^{\Lambda}$ which is affine and defined by the parameter matrix $\Lambda$.

$$\sigma(x) = \sigma^{aux}(\mathbf{q}) + \sigma^{\Lambda}(\mathbf{q}). \tag{5.15}$$

For each component $j = 1, 2, \ldots, m$, the equation $\sigma_j^{\Lambda} = 0$ is the equation of a hyperplane which is defined by $n + 1$ parameters as

$$\sigma_j^{\Lambda}(\mathbf{q}) := \lambda_{j,0} + \lambda_{j,1}\, q_1 + \ldots + \lambda_{j,n}\, q_n. \tag{5.16}$$

The half-spaces $\sigma_j^{\Lambda}(\mathbf{q}) \geq 0$ and $\sigma_j^{\Lambda}(\mathbf{q}) < 0$ are not affected by multiplying all parameters by a positive scalar, therefore we can fix one parameter, say $\lambda_{j,0}$, to be in $\{-1, 0, 1\}$. In total, for all components of the control vector, there will be $m \times (n + 1)$ parameters to choose from. By selecting the parameter matrix

$$\Lambda := \begin{bmatrix} \lambda_{1,0} & \cdots & \lambda_{1,n} \\ & \cdots & \\ \lambda_{m,0} & \cdots & \lambda_{m,n} \end{bmatrix}, \tag{5.17}$$

we define the function

$$\sigma^{\Lambda}(\mathbf{q}) = \Lambda \begin{bmatrix} 1 \\ \mathbf{q} \end{bmatrix}, \tag{5.18}$$

and therefore we define the switching function $\sigma$ as in (5.15) and the feedback law $k^{\Lambda}$ as in (5.14). Each component of the feedback law can be described as

$$k_j^{\Lambda}(\mathbf{q}) = \begin{cases} 0 & if \quad \mathbf{q} \in \Theta, \\ u_j^{\max} & if \quad \left[\sigma^{aux}(\mathbf{q}) + \Lambda \begin{bmatrix} 1 \\ \mathbf{q} \end{bmatrix}\right]_j \geq 0, \\ u_j^{\min} & if \quad \left[\sigma^{aux}(\mathbf{q}) + \Lambda \begin{bmatrix} 1 \\ \mathbf{q} \end{bmatrix}\right]_j < 0. \end{cases} \tag{5.19}$$

In our example the switching surfaces, for each control component, are planes in the state space $\mathbb{R}^3$ and can be described by 4 real parameters. The surface, and therefore the feedback, are defined by a parameter matrix $\Lambda \in \mathbb{R}^{2\times 4}$.

The feedbacks are obtained by solving, in a receding horizon strategy, the following optimal control problems with respect to matrices $\Lambda \in \mathbb{R}^{2\times 4}$.

$$\begin{array}{ll} \min\limits_{\Lambda_1,\ldots,\Lambda_{N_c}\in\mathbb{R}^{2\times 4}} & \int_t^{t+T_p} L(\mathbf{q}(s),\mathbf{u}(s))ds + W(t+T_p,\mathbf{q}(t+T_p)), \\ \text{subject to} & \dot{\mathbf{q}}\,(s) = f\,(\mathbf{q}\,(s)\,,\mathbf{u}\,(s)) \qquad \text{a.e. } s\in[t,t+T_p]\,, \\ & \mathbf{q}(t) = \mathbf{q}_t, \\ & \mathbf{q}\,(s)\in Q \qquad \text{for all } s\in[t,t+T_p]\,, \\ & \mathbf{q}\,(t+T_p)\in S, \end{array}$$

where

$$\mathbf{u}(s) = k^{\Lambda_i}(\mathbf{q}(\lfloor s\rfloor_\pi)), \qquad s\in[t+(i-1)\delta, t+i\delta), \quad i=1,\ldots,N_c, \tag{5.20}$$

$$\mathbf{u}(s) = k^{aux}(\mathbf{q}(\lfloor s\rfloor_\pi)), \qquad s\in[t+(i-1)\delta, t+i\delta), \quad i=N_c+1,\ldots,N_p. \tag{5.21}$$

This is a difficult optimization problem, since the cost is a flat function of the parameters on a large part of its domain. So, derivative-free methods were used. In this case, we used the fminsearch function of Matlab.

In conclusion, the Bang Bang feedbacks converge very fast thus, they can be applied to fast systems. Figure (5.4) shows our simulation result.

## 5.4 Notes at the end of Chapter

In this Chapter, we report results on the implementation of an Optimal Control algorithm and a stable MPC strategy for a wheeled robot, where finite parameterizations for discontinuous control functions are used, resulting in efficient computation of the control strategies. A simple case of a differential-drive mobile robot moving on a plane was explored.

We distinguish if we are looking into an open-loop control solution or a closed-loop control solution to the Optimal Control Problem.

For an open-loop solution we apply (i) undirect methods or (ii) direct methods. We developed an implementation of the direct method when the control functions are approximated by piecewise constant function and polynomials, respectively.

Regarding closed-loop optimal control a feedback law might be obtained by (i) Dynamic Programming / Hamilton Jacobi Equation, (ii) by restricting the class of feedback functions (e.g., affine, bang-bang) and thereby obtain an approximate solution, or (iii) by using MPC, that generates a feedback by solving a sequence of open-loop OCPs and thereby also obtaining an approximate solution.

We obtained a closed-loop optimal control feedback law by restricting the class of feedback functions.

Here we summarize the optimal control algorithms but the contribution is in its implementation.

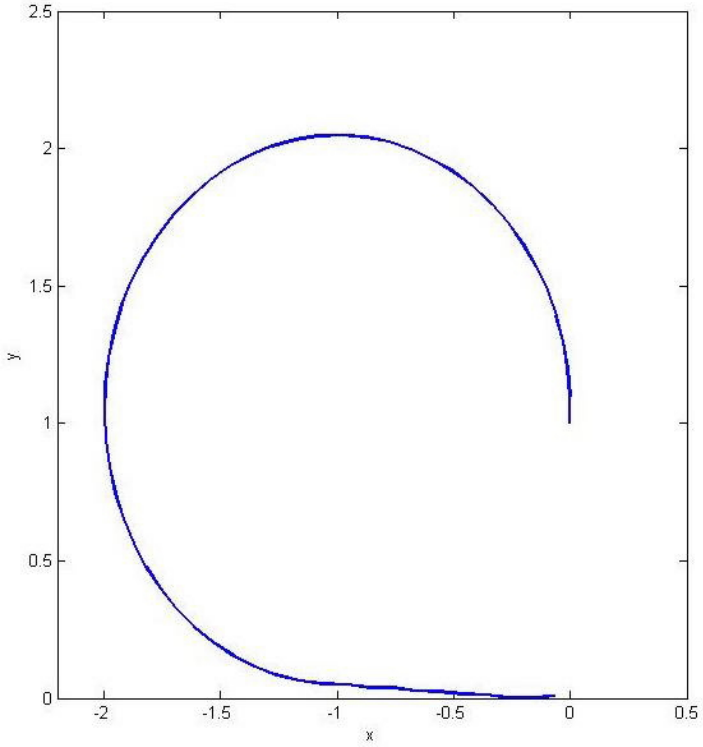


Figure 5.4: Trajectory in the plan using Bang-Bang feedbacks.

This work was presented in a poster at the 1st Porto meeting on Mathematics for Industry in 2009 at Faculdade de Ciências da Universidade do Porto.

# Chapter 6

# Application of the MPC strategy to nonholonomic vehicles

## 6.1 Introduction

The aim of this Chapter is to address the control problem for nonholonomic wheeled mobile robot moving on the plane. The possible motion tasks can be classified as follows [LOS98]:

- Point-to-point motion: a desired goal configuration must be reached starting from a given initial configuration.
- Path following: the robot must reach and follow a geometric path in the cartesian space starting from a given initial configuration (on or off the path).
- Trajectory tracking: a reference point on the robot must reach and follow a trajectory in the Cartesian space (i.e., a geometric path with an associated timing law) starting from a given initial configuration (on or off the trajectory).

The specific robotic system considered is the differential-drive mobile robot car (see Figure 2.4). The configuration of this robot is represented by the position and orientation of its main body in the plane, and by the angle of the steering wheels. Two velocity inputs are available for motion control.

This system is nonholonomic and is controllable, as it has been shown in Chapter 2.

## 6.2 Point-to-point motion

### 6.2.1 Introduction

The *point-to-point motion* of nonholonomic mobile robots (also known as *regulating* or *stabilizing nonholonomic* mobile robots) consists to move the robot towards a goal position from any initial

position. We aim to drive the state (or position) of our system to a pre-specified point (usually the origin is considered, without loss of generality). A parking manoeuver is an example of such application in wheeled vehicles (see Figure 2.1).

As already seen in Chapter 2, the linearized models of nonholonomic systems are not controllable, and continuous time-invariant feedback control does not exist in the regulation problem [Bro83]. A number of control strategies have been proposed to handle this problem, for example, Lyapunov control [Ast95], dynamic feedback linearization [OLV02], etc.

The *point-to-point motion* of nonholonomic mobile robots is a complex problem [OLV02]. In particular because there is no continuous time-invariant feedback to stabilize the system, and the linearization around some operating points (the case of wheeled vehicles with zero velocity, for example, is not stabilizable). Nevertheless, this problem has been addressed by MPC, see e.g., [VEN01] and [Fon01].

MPC is one of the frequently applied advanced control techniques in industry, which is designed to handle optimization problems with constraints. Due to the use of a predictive control horizon, the control stability becomes one of the main problems. An MPC framework with guaranteed stability properties addressing a general class of nonlinear systems is discussed in [Fon03] and [FM03a]. In the following we are going to present the results on the implementation of a stabilizing MPC strategy. Conditions under which steering to a set is guaranteed are established.

### 6.2.2 Problem formulation

In the point-to-point motion, the robot has to move from an initial to a final position in a given time $t_f$ (see Figure 2.11). The algorithm should generate a trajectory which is also capable of optimizing some performance criterion when the point is moved from one position to another.

The point-to-point motion task is a stabilization problem for an (equilibrium) point in the robot state space. For the differential-drive mobile robot car (see Figure 2.4), two control inputs are available for adjusting three configuration variables, namely the two cartesian coordinates characterizing the position of a reference point on the vehicle and its orientation.

Consider a differential-drive mobile robot moving on a plane, and represented by the following kinematic model

$$\begin{cases} \dot{x} &= v\cos\theta, \\ \dot{y} &= v\sin\theta, \\ \dot{\theta} &= w, \end{cases} \tag{6.1}$$

or in a compact form:

$$\dot{\mathbf{q}}(t) = \mathbf{f}\left(\mathbf{q}(t), \mathbf{u}(t)\right), \tag{6.2}$$

where $\mathbf{q} = [x \;\; y \;\; \theta]^T$ e $\mathbf{u} = [v \;\; w]^T$.

The coordinates $(x, y)$ are the position in the plane of the midpoint of the axle connecting the rear wheels, and $\theta$ denotes the heading angle measured from the $x$-axis. The controls $v$ and $w$ are the linear and angular velocity, respectively.

The objective is to drive this system to the origin. In practice we consider a target set

$$\Theta = \left\{ \mathbf{q} = (x, y, \theta) \in \mathbb{R}^2 \times [-\pi, \pi] : \|(x, y)\| \leq \varepsilon_1, |\theta| \leq \varepsilon_2 \right\},$$

where $\varepsilon_1, \varepsilon_2 > 0$.

The feedback control is obtained by repeatedly solving online open-loop optimal control problems (OCP) at each sampling instant $t_i$, every time using the current measure of the state of the plant $\mathbf{x}_{t_i}$.

The MPC control law is obtained with the algorithm already described in Chapter 4:

1. Measure the state of the plant $\mathbf{q}_{t_i}$;
2. Get $\overline{u} : [t_i, t_i + T] \to \mathbb{R}^n$ the solution of the OCP:

$$\begin{array}{ll} \text{Minimize} & \int_{t_i}^{t_i+T} L\left(\mathbf{q}(t), \mathbf{u}(t)\right) dt + W\left(\mathbf{q}\left(t_i + T\right)\right), \\ \text{subject to:} & \dot{\mathbf{q}}(t) = f\left(\mathbf{q}\left(t\right), \mathbf{u}\left(t\right)\right), \ \text{ a.e. } t \in [t_i, t_i + T], \\ & \mathbf{q}\left(t_i\right) = \mathbf{q}_{t_i}, \\ & \mathbf{u}\left(t\right) \in U\left(t\right), \ \text{ a.e. } t \in [t_i, t_i + T], \\ & \mathbf{q}\left(t_i + T\right) \in S; \end{array} \tag{6.3}$$

3. Apply to the plant the control $\mathbf{u}^*\left(t\right) := \overline{\mathbf{u}}\left(t\right)$ in the interval $[t_i, t_i + \delta[$. The remaining control, $\overline{\mathbf{u}}\left(t\right), t > t_i + \delta$, discarded;
4. Repeat the procedure from (1.) for the next sampling instant, $t_i = t_i + \delta$, using the new measure of the state of the plant $\mathbf{q}_{t_i+1}$.

The pair $(\overline{\mathbf{q}}, \overline{\mathbf{u}})$ denotes an optimal solution to an open-loop OCP. The process $(\mathbf{q}^*, \mathbf{u}^*)$ is the closed-loop trajectory and control resulting from the MPC strategy. The *design parameters* (the variables present in the open-loop OCP that are not from the system model) are: the time horizon $T$, the functions $L$ and $W$, and the set $S$. They must satisfy the stability conditions (**SC**) and therefore guarantee that the resulting MPC strategy is stabilizing, and are constructed based on a simple strategy that drives the system to the origin.

Consider the stability condition **SC** in the Chapter 4 [Fon03]. Next, we analyse the selection of design parameters guaranteeing stability for the differential-drive mobile robot car.

Supose that the target set is (consider $\varepsilon_1 = 0.05$ and $\varepsilon_2 = 0.1$):

$$\Theta = \left\{ \mathbf{q} = (x, y, \theta) \in \mathbb{R}^2 \times [-\pi, \pi] : \|(x, y)\| \leq 0.05, |\theta| \leq 0.1 \right\}.$$

A possible stabilizing strategy (not necessarily the best) might be performing the following actions:

**A.** Rotate the robot until its heading angle $\theta$ points towards to the origin of the plane $(x, y) = (0, 0)$;

**B.** Move the robot until the midpoint $(x, y)$ of the axle connecting the rear wheels satisfies $\|(x, y)\| \leq 0.05$;

**C.** Rotate the robot until its heading angle $\theta$ is aligned with the $x$ axis, that is, $\theta = 0$.

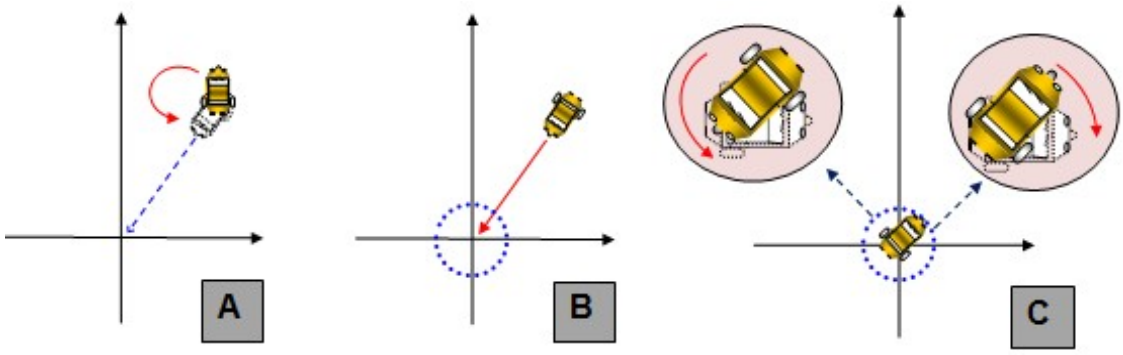


Figure 6.1: A stabilizing strategy.

Actually, the Actions **A.** and **B.** are carried out simultaneously.

It is convenient to define $\phi(x, y)$ to be the angle in $[-\pi, \pi]$ that points to the origin from position $(x, y)$ away from the origin, more precisely:

$$\phi(x,y) = \begin{cases} 0 & \text{se} \quad \|(x,y)\| \leq \varepsilon, \\ -(\pi/2)\,\text{sign}\,(y) & \text{se} \quad \|(x,y)\| > \varepsilon, x = 0, y \neq 0, \\ \tan^{-1}(y/x) + \pi & \text{se} \quad \|(x,y)\| > \varepsilon, x > 0, \\ \tan^{-1}(y/x) & \text{se} \quad \|(x,y)\| > \varepsilon, x < 0. \end{cases} \tag{6.4}$$

This function is easily implementable in Matlab (which have a four-quadrant inverse tangent built-in function) as $\phi(x, y) = ATAN2(-y, -x)$.

Considering all this, the control function $\mathbf{k}^{aux}$, is:

$$\begin{aligned} \text{if } \|(x,y)\| < 0.05, \quad & \text{then} \\ & v = 0 \text{ and } w = -\tfrac{\pi}{4} \times \theta, \\ & \text{else} \\ & v = 0.1 \text{ and } w = -\tfrac{\pi}{4} \times [\theta - \phi(x,y)]\,. \end{aligned} \tag{6.5}$$

Define the design parameters to be:

**Terminal set $S$:**

Define the terminal set $S$ to be the set of states heading towards the origin of the plane together with the origin of the plane, that is:

$$S = \left\{ \mathbf{q} = (x, y, \theta) \in \mathbb{R}^2 \times [-\pi, \pi] : (x, y, \theta) \in \Theta \vee (x, y) = (0, 0) \right\}. \tag{6.6}$$

**The time horizon $T$:**

The set $S$ can be reached from any state if we define the horizon to be the time to complete an 180 degrees (Action **A**) action turn, that is

$$T = \pi / w_{\max} = \pi. \tag{6.7}$$

**The functions $L$ and $W$:**

Consider the functions $L(\mathbf{q})$ and $W(\mathbf{q})$ used in [FM03a]

$$L(\mathbf{q}) = x^2 + y^2 + \theta^2 \tag{6.8}$$

and

$$W(\mathbf{q}) = \frac{1}{3}\left(r^3 + |\theta|^3\right) + r\theta^2, \tag{6.9}$$

where

$$r = \sqrt{x^2 + y^2}.$$

The set $S$ is closed and contains the origin. Also, the structure of the strategy guarantees that the controls generated are always within $U$.

The function $M$, whose existence is assumed, can simply be made equal to the function $L$. It is easily checked that $M$ and $L$ satisfy the conditions in **(SC2)**. Actually, any quadratic running cost of the type $L(\mathbf{q}, \mathbf{u}) = \mathbf{q}^T\mathbf{Q}\mathbf{q} + \mathbf{u}^T\mathbf{R}\mathbf{u}$ (with $Q > 0$ and $R > 0$) is always a possible choice for $L$ satisfying **(SC2)** with $M = \mathbf{q}^T\mathbf{Q}\mathbf{q}$.

The function $W$ is Lipschitz continuous and positive semi-definite as required.

It is trivial to prove that these design parameters satisfy conditions (**SC1**) to (**SC4**) and (**SC5b**). We verify (**SC5a**) below.

$$\nabla W(\mathbf{q}) = \left(xr + x\theta^2/r, yr + y\theta^2/r, \theta|\theta| + 2\theta r\right),$$

when $\theta \neq 0$ and $r \neq 0$ (i.e., $(x, y) \neq (0, 0)$).

1. Suppose $\|(x, y)\| \leq 0.05$ and consider the following cases:

   (a) if $|\theta| \leq 0.1$, then the robot is already in the target set $\Theta$.

(b) if $|\theta| > 0.1$, and since $r \leq 0.05 \leq 0.1$ then $|\theta| > r$. The control strategy is rotate the robot until its heading angle $\theta$ is aligned with the $x$ axis, that is, the controls are chosen as $v = 0$ and $w = -\frac{\pi}{4} \times \theta$, therefore,

$$f(\mathbf{q}, \mathbf{u}) = \left(0, 0, -\frac{\pi}{4} \times \theta\right).$$

so,

$$\begin{aligned} \nabla W(\mathbf{q}) \cdot f(\mathbf{q}, \mathbf{u}) &= -\tfrac{\pi}{4} \times \theta \left(\theta |\theta| + 2\theta r\right) \\ &= -\tfrac{\pi}{4} \times \left(|\theta| + 2r\right) \theta^2 \\ &\leq -\tfrac{\pi}{4} \times 3 |\theta| L(\mathbf{q}) \\ &\leq -L(\mathbf{q}). \end{aligned}$$

2. Suppose $\|(x, y)\| > 0.05$. The control strategy is to rotate the robot until the heading angle points to the origin and then move it to the origen. The controls are chosen as $v = 0.1$ and $w = -\frac{\pi}{4} \times [\theta - \phi(x, y)]$. Let $a = \theta - \phi(x, y)$. Then

$$\begin{aligned} f(\mathbf{q}, \mathbf{u}) &= \left(0.1 \cos\theta, 0.1 \sin\theta, -\frac{\pi}{4} a\right) \\ &= \left(\frac{\cos\phi \cos a - \sin\phi \sin a}{10}, \frac{\sin\phi \cos a + \cos\phi \sin a}{10}, -\frac{\pi}{4} a\right). \end{aligned}$$

Using the fact that

$$\tan^{-1} z = \sin^{-1}\left(z/\sqrt{1+z^2}\right) = \cos^{-1}\left(1/\sqrt{1+z^2}\right),$$

we have, in the case $x > 0$, that (note that, $a \leq \frac{\pi}{2}$)

$$\cos\phi = \cos\left[\tan^{-1}\left(\frac{y}{x}\right) + \pi\right] = -\frac{x}{\sqrt{x^2+y^2}}$$

and

$$\sin\phi = \sin\left[\tan^{-1}\left(\frac{y}{x}\right) + \pi\right] = -\frac{y}{\sqrt{x^2+y^2}}.$$

Thus,

$$f(\mathbf{q}, \mathbf{u}) = \left(\frac{1}{10}\left(-\frac{x}{r}\cos a + \frac{y}{r}\sin a\right), \frac{1}{10}\left(-\frac{y}{r}\cos a - \frac{x}{r}\sin a\right), -\frac{\pi}{4} a\right).$$

If $\theta \neq 0$ the gradient of $W$ is well defined and (note that $2r\theta \leq r^2 + \theta^2$)

$$\begin{aligned} \nabla W(\mathbf{q}) \cdot f(\mathbf{q}, \mathbf{u}) &= -\tfrac{1}{10}\left(x^2 + y^2 + \theta^2\right) \cos a - \tfrac{\pi}{4} a \ \left(\theta |\theta| + 2\theta r\right) \\ &\leq -0.1\left(x^2 + y^2 + \theta^2\right) - \tfrac{\pi^2}{8}\left(\theta^2 + r^2 + \theta^2\right) \\ &= -\left(0.1 + \tfrac{\pi^2}{8}\right) L(\mathbf{q}) \leq -L(\mathbf{q}). \end{aligned}$$

The cases when $x < 0$ and $x = 0$ can be verified in a similar way.

The condition (**SC5b**), states that the trajectory does not leave the set S immediately. For a suficiently small $\delta$ (let $\delta = 0.1$), we have

$$\mathbf{q}\left(t+\delta; t, \mathbf{q}_t, \mathbf{k}^{aux}\right) \in S.$$

In conclusion, as this choice of design parameters satisfy the stability condition **SC** in the Chapter 4 [Fon03], the feedback controller resulting from the application of the MPC strategy is a stabilizing controller.

### 6.2.3 Implementation and simulation

The optimal control problem 6.3 (in the Bolza form) is reformulated in an equivalent Mayer form and discretized. The resulting nonlinear programmes are then solved in MATLAB using as initial guess the auxiliary stabilizing strategy. Simulation results are shown both using the auxiliary stabilizing strategy as well as the MPC strategy for the initial position $\mathbf{q}_0 = \left(0, 1, \frac{\pi}{2}\right)^T$.

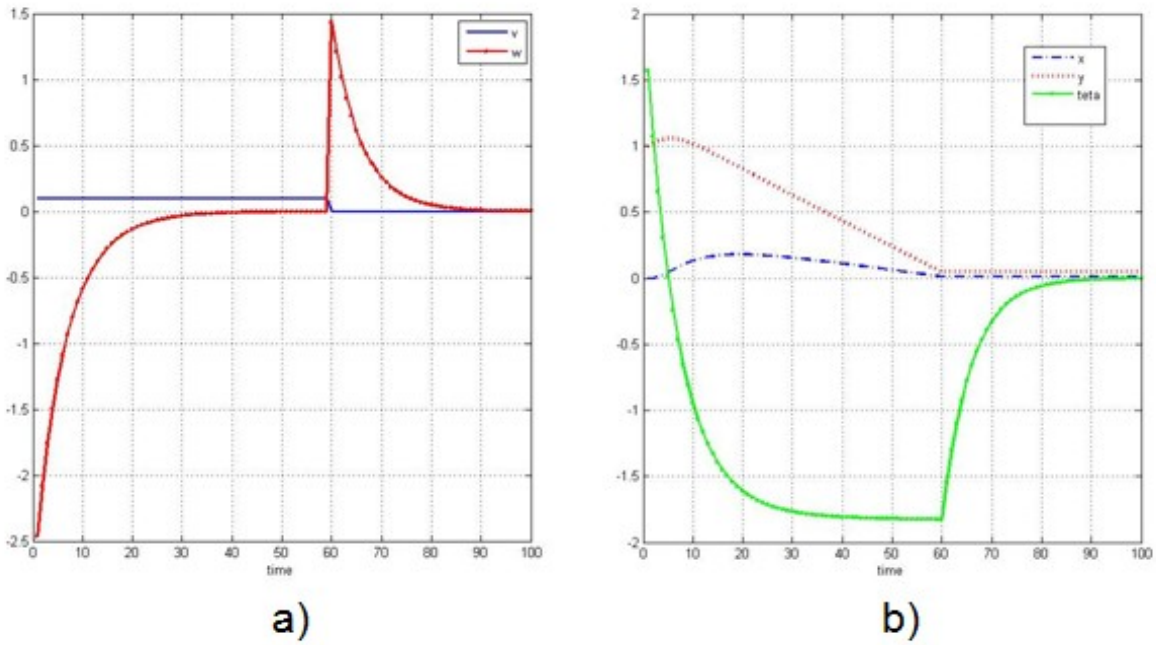


Figure 6.2: a) Auxiliary stabilizing strategy $\mathbf{k}^{aux}$; b) Trajectory using $\mathbf{k}^{aux}$.

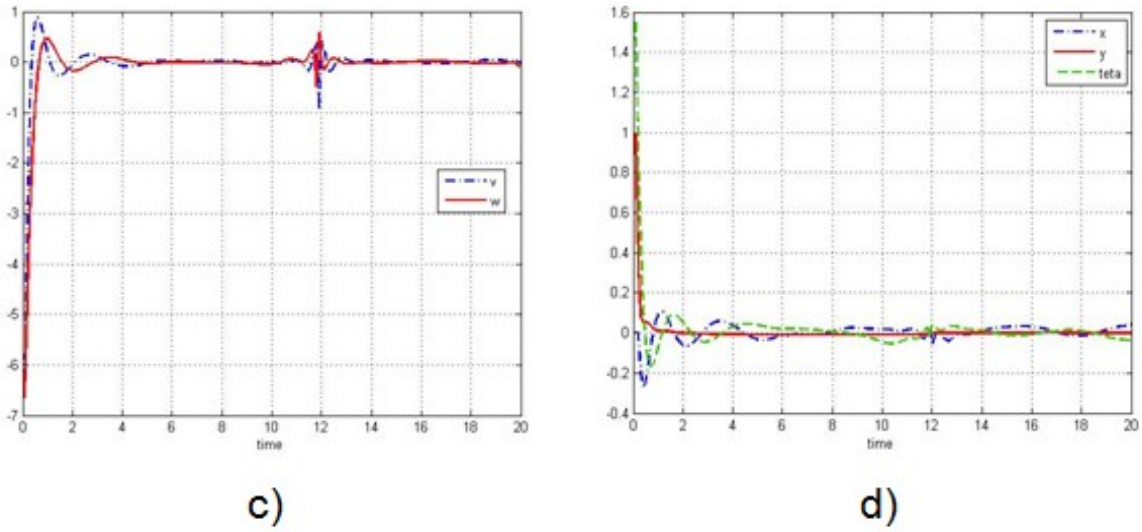


Figure 6.3: c) MPC control; d) MPC Trajectory.

By observing the Figure (6.3-c) it can be seen that the robot reaches the set $\Theta$ very quickly, that is, after 6 seconds the position $\mathbf{q} = (x, y, \theta)$ of the robot satisfies de conditions $\|(x, y)\| \leq 0.05$ and $|\theta| \leq 0.1$. Due to the complicated numerical computations, in the solution for $t = 12$ there are small "numerical gaps".

## 6.3 MPC for Path-following of Nonholonomic Systems

### 6.3.1 Introduction

Trajectory-tracking in some nonholonomic systems, such as wheeled vehicles, is much simpler than the point-to-point motion [GH05, GH06]. If we assume we are tracking a dynamical feasible trajectory, then, some wheeled vehicles behave (for nonzero velocities) as if they were holonomic. Consider the example a car-like vehicle performing a parking or an overtaking manoeuver as discussed in [FFC09]. See figures 2.1 in Chapter 2 and 6.4 below.

When considering path-following instead of trajectory-tracking, the degree of freedom on how fast we move along the path can be used to increase performance in certain systems [AHK05].

MPC can explore effectively this degree of freedom and in addition it can deal naturally with the constraints omnipresent in path-following problems. MPC is known to be a technique that deals appropriately and explicitly with constraints. In fact, many researchers argue that the capacity of dealing naturally and effectively with constraints is the main reason for the industrial success of MPC; e.g. [May01, Mac02, QB03, GSD04].

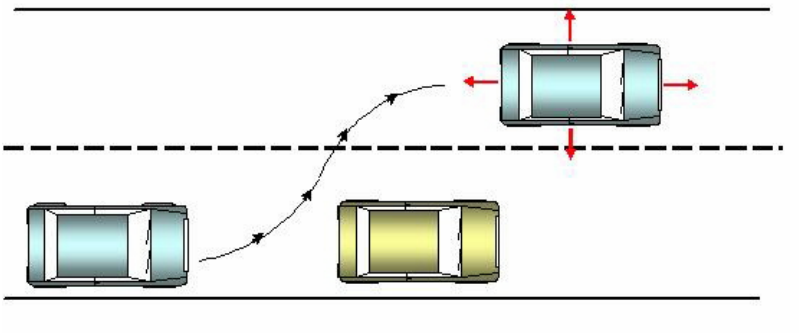

Figure 6.4: Car in an overtaking manoeuver: can move in all directions relative to the other car.

We address the path-following problem by converting it into a trajectory -tracking problem and determine the speed profile at which the path is followed inside the optimization problems solved in the MPC algorithm.

The MPC framework will solve a sequence of optimization problems that will find an initial point, a speed profile, and a feedback control to track the trajectory of a virtual reference vehicle, i.e., the MPC framework will find a feedback control to follow the path given.

The method is illustrated in a simple differential-drive mobile robot.

### 6.3.2 The MPC Framework

### 6.3.3 Model predictive control and path-following

Consider a 2D path[1] defined in a parameterized form as

$$x_d(\tau) : [0, T_d] \to \mathbb{R}, \tag{6.10}$$
$$y_d(\tau) : [0, T_d] \to \mathbb{R}. \tag{6.11}$$

We will define a positive function of time, the speed $\nu$ at which the path is followed, such that

$$\dot{\tau} = \nu(t) \quad a.e.,$$

and define a trajectory for a virtual reference vehicle

$$t \mapsto x_r(t) = x_d(\tau(t)), \tag{6.12}$$
$$t \mapsto y_r(t) = y_d(\tau(t)). \tag{6.13}$$

[1] The 2D case is described here for the sake of simplicity. The 3D case is defined in a similar way.

If we consider that the path is followed at constant speed $\nu(t) = \nu$ and the virtual reference vehicle starts following the path at $(x_d(\tau_0), y_d(\tau_0))$, then

$$\tau(t) = \tau_0 + \nu t.$$

When we consider a wheeled vehicle with the non-slipping nonholonomic constraint (a unicycle or a car-like vehicle), the heading angle is defined by the velocity vector (uniquely if we assume forward velocity) as

$$\theta(\tau) = ATAN2\ (\dot{y}_d, \dot{x}_d) + 2k\pi, \qquad k \in \mathbb{Z}.$$

The path-following problem can be converted into a tracking problem as follows:

1. Find an initial point $Q_0$ in the path to start the trajectory of the virtual reference vehicle. Given the initial (current) position of our vehicle $(x_0, y_0)$, find the path parameter value that corresponds to a point in the path that is the closest to $(x_0, y_0)$.

   $$\tau_0 = \underset{\tau \in [0, T_d]}{\operatorname{argmin}} \|(x_d(\tau), y_d(t)) - (x_0, y_0)\|.$$

   See Figure 6.5.

2. Select a speed profile at which the path is to be followed

   $$\nu : [0, T] \mapsto \mathbb{R}_+.$$

3. Find a feedback control to track the trajectory of the virtual reference vehicle

   $$t \mapsto (x_r(t), y_r(t)).$$

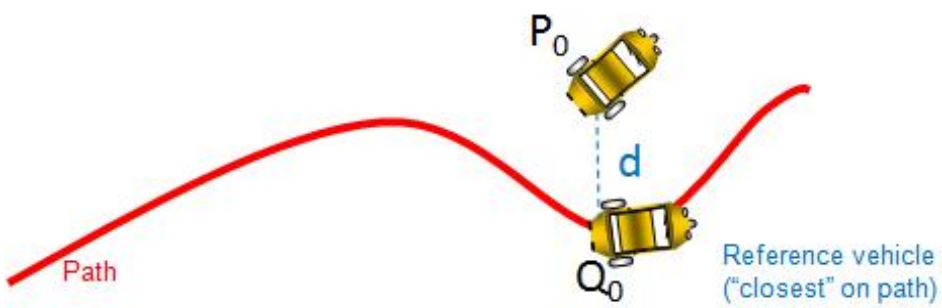


Figure 6.5: Find an initial point $Q_0$ in the path that is the closest to the current position.

This problem can be solved in a MPC framework. That is, the MPC framework will solve a sequence of optimization problems that will find:

- The initial point $Q_0$;
- A speed profile $\nu$;
- A feedback control to track the trajectory of the virtual reference vehicle,

That is, the MPC framework will find a feedback control to follow the path given.

We repeatedly solve the following open-loop optimal control problems at each sampling instant $t_i \in \pi$, every time using the current measure of the state of the plant $\mathbf{q}_{t_i} = (x_{t_i}, y_{t_i}, \theta_{t_i})$.

Minimize

$$\int_0^{T_d/\nu} (\|\mathbf{q}(t) - \mathbf{q}_d(\tau(t))\|_Q^2 + \|\mathbf{u}(t)\|_R^2)dt + \|\mathbf{q}(T_d/\nu)\|_P^2, \tag{6.14}$$

subject to:

$$\begin{aligned} &\dot{\mathbf{q}}(t) = \mathbf{f}(t, \mathbf{q}(t), \mathbf{u}(t)) && a.e. \quad t \in [0, T_d/\nu] \\ &\mathbf{q}(0) = \mathbf{q}_{t_i}, \\ &\tau_0 = \underset{\tau \in [0,T_d]}{\operatorname{argmin}} \|\mathbf{q}_d(\tau) - \mathbf{q}_{t_i}\|, \\ &\tau(t) = \tau_0 + \nu t, \\ &\mathbf{u}(t) \in U, && \forall t \in [0, T_d/\nu]. \end{aligned} \tag{6.15}$$

Path-following has been addressed by an MPC framework in [FF08a]. There a stability analysis is carried out.

### 6.3.4 Application to a unicycle-type mobile robot

Consider a differential-drive mobile robot moving on a plane, a unicycle -type as in Figure 2.4 of Chapter 2, and represented by the kinematic model (2.17) with $\theta(t) \in [-\pi, \pi]$ and $v, w \in [-1, 1]$.

The coordinates $(x, y)$ are the position in the plane of the midpoint of the axle connecting the rear wheels, and $\theta$ denotes the heading angle measured from the $x$-axis. The controls $v$ and $w$ are the linear and angular velocity, respectively.

Our problem is to make this vehicle follow the eight-shaped path

$$\begin{aligned} x_d(\tau) &= \sin(\tau/10), && \tau \in [0, 38\pi], \\ y_d(\tau) &= \sin(\tau/20), && \tau \in [0, 38\pi]. \end{aligned}$$

By selecting a positive speed function $\nu$ and an initial point in the path we define the trajectory for our virtual reference vehicle

$$\begin{aligned} x_r(t) &= \sin(\tau(t)/10), \\ y_r(\tau) &= \sin(\tau(t)/20), \\ \tau(t) &= \tau_0 + \textstyle\int_0^t \nu(s)ds. \end{aligned}$$

We could now follow some nonlinear MPC approaches to this problem, such as [VEN01], or approaches for which stability was rigourously established [Fon02b, GH05, GH06]. But,

our knowledge of this problem, enable us to address small deviations of the nominal path by a point-wise linearized model, which make optimization problems easier and more efficiently solved.

Since this trajectory is twice differentiable and the velocity is always nonzero, we can define

$$\theta_r(t) = ATAN2\ (\dot{y}_r, \dot{x}_r) + 2k\pi, \qquad k \in \mathbb{Z}.$$

The open-loop control laws, that would guide our virtual reference vehicle along the path given, can be explicitly computed as [OLV02]

$$v_r\,(t) = \pm\sqrt{\dot{x}_d^2\,(t) + \dot{y}_d^2\,(t)} \tag{6.16}$$

and

$$w_r(t) = \frac{\ddot{y}_d(t)\dot{x}_d(t) - \ddot{x}_d(t)\dot{y}_d(t)}{\dot{y}_d^2(t) + \dot{x}_d^2(t)}, \qquad a.e.\ . \tag{6.17}$$

In our regulation problem, we want to drive to zero

$$\mathbf{q} - \mathbf{q}_r = (x, y, \theta) - (x_r, y_r, \theta_r).$$

If we define this error rotated to a frame aligned with the reference trajectory as

$$e = \mathcal{R}_\theta(\mathbf{q} - \mathbf{q}_r) \tag{6.18}$$

where

$$\mathcal{R}_\theta = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

The linearized error will satisfy

$$\dot{e} = \begin{bmatrix} o & w_r & 0 \\ -w_r & 0 & v_r \\ 0 & 0 & 0 \end{bmatrix} e + \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \tag{6.19}$$

where

$$\begin{aligned} u_1 &= -v + v_r \cos e_3, \\ u_2 &= w_r - w. \end{aligned} \tag{6.20}$$

We are now faced with a linear time-varying model, but which is valid only on a small local neighborhood, and with input constraints. An ideal setting for MPC.

We determine $u_1$ and $u_2$ by MPC and then recover $v$ and $w$ by (6.20). Applying such controls to the system

$$\begin{cases} \dot{x}\,(t) &= (1+\eta_1)\,v \cdot \cos\theta\,(t) + \eta_4, \\ \dot{y}\,(t) &= (1+\eta_2)\,v \cdot \sin\theta\,(t) + \eta_5, \\ \dot{\theta}\,(t) &= (1+\eta_3)\,w, \end{cases}$$

where $\eta_i$, $i = 1 \ldots 5$ are white noise sequences with variance between 0.01 and 0.05, results in the trajectory depicted in Figure 6.6. The Optimal Control Problems solved in the MPC setting are as follows:

Minimize

$$\int_0^{T_d/\nu} (\|q(t) - q_d(\tau(t))\|_Q^2 + \|u(t)\|_R^2)dt + \|q(T_d/\nu)\|_P^2, \tag{6.21}$$

subject to:

$$\begin{array}{ll} \dot{e}(t) = Ae(t) + Bu(t), & a.e.t \in [0, T_d/\nu], \\ e(0) = \mathcal{R}_\theta(q_d(\tau_0) - q(t_i)), & \\ \tau_0 = \text{argmin}_{\tau \in [0,T_d]} \|q_d(\tau) - q_{t_i}\|, & \\ \tau(t) = \tau_0 + \nu t, & \forall t \in [0, T_d/\nu], \\ |u_1(t)| \leq U_{max}, & \forall t \in [0, T_d/\nu], \\ |u_2(t)| \leq U_{max}, & \forall t \in [0, T_d/\nu], \end{array} \tag{6.22}$$

where $A$ and $B$ are as in given equation (6.19), $Q = diag(1, 1, 0.1)$, $R = diag(0.10.1)$, and $P$ solves the algebraic Riccati equation

$$A'P + PA - PBR^{-1}B'P + Q = 0,$$

imposing this way, a terminal cost that is known to guarantee stability [CA98b].

## 6.4 Notes at the end of Chapter

In this Chapter results are reported on the implementation of a stabilizing MPC strategy to a wheeled robot. Conditions under which steering to a set is guaranteed are established. A set of design parameters satisfying all these conditions for the control of a unicycle mobile robot are derived.

We discuss the use of MPC to address the problem of path-following of nonholonomic systems. We argue that MPC can solve this problem in an effective and relatively easy way, and has several advantages relative to alternative approaches.

We address the path-following problem by converting it into a trajectory-tracking problem and determine the speed profile at which the path is followed inside the optimization problems solved in the MPC algorithm.

The MPC framework solves a sequence of optimization problems that find an initial point, a speed profile, and a feedback control to track the trajectory of a virtual reference vehicle, i.e., the MPC framework will find a feedback control to follow the path given.

The method is illustrated in a simple differential-drive mobile robot.

The material in this Chapter has been published in [CF10].

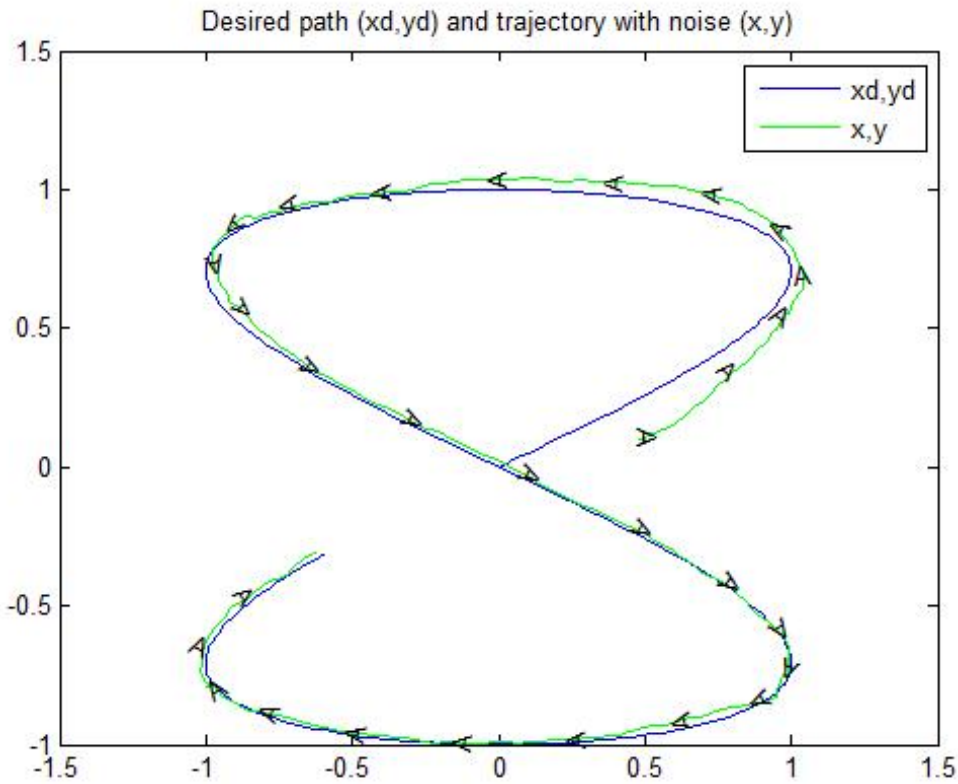


Figure 6.6: Trajectory of unicycle-type mobile robot with the MPC controller.

# Chapter 7

# Control of vehicle formation

## 7.1 Introduction

Over the last few years, formation control of multiple vehicles has received a lot of attention from the control community. Compared with a single mobile robot, many advantages of mobile robots working together have been shown in many tasks, such as surveillance, forest fire detection, search missions, automated highways, exploration robots, among many others (see e.g., [Mur07]). However many issues arise in multi-robot systems, such as communication structures, task allocation, decentralization/centralization, etc.

Formation control is an important technique to achieve cooperative behavior in multi-robot systems. The task is to control a group of mobile robots to follow a predefined path or trajectory, while maintaining a desired formation pattern. Research on formation control also helps people to better understand some biological social behaviors, such as swarm of insects and flocking of birds.

In this Chapter we propose a control scheme for a set of vehicles moving in a formation.

The control methodology selected is a two-layer control scheme where each layer is based on Model Predictive Control (MPC). Control of multi-vehicle formations and/or distributed MPC schemes have been proposed in the literature. See the recent works [DM06, RMS07] and the references therein.

The reason why two-layers are used in the control scheme is because there are two intrinsically different control problems:

- **The trajectory control problem**: devise a trajectory, and corresponding actuator signals, for the formation as a whole;

- **Maintain the formation**: change the actuator signals in each vehicle to compensate for small changes around a nominal trajectory and maintain the relative position between vehicles.

These control problems are intrinsically different because, on the one hand, most vehicles (cars, planes, submarines, wheeled vehicles) are nonholonomic (cannot move in all directions). On the other hand, while the vehicles are in motion, the relative position between them in a formation can be changed in all directions (as if they were holonomic).

As an example consider a vehicle whose dynamics we are familiar with: a car. Consider the car performing a parking maneuver or performing an overtaking maneuver. See Figures 2.1 in Chapter 2 and 6.4 in Chapter 6.

As it has already been said in Chapter 2, in the first situation, we are limited by the fact that the car cannot move sideways: it is nonholonomic. It is a known result that we need a nonlinear controller, allowing discontinuous feedback laws, to stabilize this system in this maneuver [SS80, Bro83].

In the second maneuver, the vehicle is in motion and small changes around the nominal trajectory can be carried out in all directions of the space. In an overtaking maneuver we can move in all directions relative to the other vehicle. This fact simplifies considerably the controller design. In fact, a linear controller is an appropriate controller to deal with small changes in every spatial direction around a determined operating point.

For a different approach that also decouples the path following from the inter-vehicle coordination problem see [GPSK07].

The other option, here is to design and use controllers based on the Model Predictive Control technique, supported by several reasons

As already said in Chapter 4, MPC is known to be a technique that deals appropriately and explicitly with constraints. In fact, many researchers argue that the capacity of dealing naturally and effectively with constraints is the main reason for the industrial success of MPC (see e.g., [May01, Mac02, QB03]).

In the problem of controlling a vehicle formation, the constraints on the trajectory are an important problem characteristic:

- To avoid collisions between the vehicles in the formation;
- To avoid obstacles in the path;
- and, for other reasons of safety or reliability of operation (e.g., minimum altitude or velocity in a plane).

Therefore, constraints should be appropriately dealt with. A possible way to deal with constraints is to use unconstrained design methods combined with a "cautious" approach of operating far from the constraints (e.g., imposing large distances between the vehicles in the formations). But, such procedures would, in general, reduce the performance that would be achievable by operating closer to the limits [GSD04].

Another possible approach is to push the trajectory away from the constraints by penalizing a vehicle close to an obstacle or other vehicle (e.g., potential field approaches, or other

optimization-based approaches that do not impose explicitly the constraints), but these methods do not guarantee that the constraints will be satisfied.

Here, we use MPC imposing explicit constraints both on the vehicle inputs and on the vehicle trajectory.

Another advantage of MPC is that, because MPC is an optimization-based method, it has desirable performance properties according to the performance criteria defined. In addition, it has intrinsic robustness properties [MS97].

The major concern in using such a technique is often the need to solve optimization problems in real-time which might be difficult for fast systems. However, there are recent results on particular parameterizations for the optimal control problems involved that allow MPC to address fast systems (see e.g., [Ala06b]). We shall see that the control strategy proposed appropriately deals with this problem.

## 7.2 The MPC Framework

Consider a nonlinear plant with input and state constraints, where the evolution of the state after time $t_0$ is predicted by the following model.

$$\begin{array}{ll} \dot{\mathbf{q}}(s) = f(s, \mathbf{q}(s), \mathbf{u}(s)), & \text{a.e. } s \geq t_0, \\ \mathbf{q}(t_0) = \mathbf{q}_{t_0} \in Q_0, & \\ \mathbf{q}(s) \in Q \subset \mathbb{R}^n, & \text{for all } s \geq t_0, \\ \mathbf{u}(s) \in U, & \text{a.e. } s \geq t_0. \end{array}$$

The data of this model comprise a set $Q_0 \subset \mathbb{R}^n$ containing all possible initial states at the initial time $t_0$, a vector $\mathbf{q}_{t_0}$ that is the state of the plant measured at time $t_0$, a given function $f : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$, and a set $U \subset \mathbb{R}^m$ of possible control values.

We assume this system to be asymptotically controllable on $Q_0$ and that for all $t \geq 0$ $f(t, 0, 0) = 0$. We further assume that the function $f$ is continuous and locally Lipschitz with respect to the second argument.

The construction of the feedback law can be accomplished by using a sampled-data MPC strategy [FMG07]. Consider a sequence of sampling instants $\pi := \{t_i\}_{i \geq 0}$, with a constant inter-sampling time $\delta > 0$ such that $t_{i+1} = t_i + \delta$ for all $i \geq 0$. Consider also the control horizon and predictive horizon, $T_c$ and $T_p$, with $T_p \geq T_c > \delta$, and an auxiliary control law $k^{aux} : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^m$. The feedback control is obtained by repeatedly solving online open-loop optimal control problems $P(t_i, \mathbf{q}_{t_i}, T_c, T_p)$ at each sampling instant $t_i \in \pi$, every time using the current measure of the state of the plant $\mathbf{q}_{t_i}$.

$$
\begin{aligned}
P(t, \mathbf{q}_t, T_c, T_p) \quad : & \\
& \text{Minimize} \quad \int_t^{t+T_p} L(s, \mathbf{q}(s), \mathbf{u}(s))ds + W(t+T_p, \mathbf{q}(t+T_p)), && (7.1)\\
& \text{subject to}: \quad \dot{\mathbf{q}}(s) = f(s, \mathbf{q}(s), u(s)), \quad \text{a.e. } s \in [t, t+T_p], \\
& \qquad \mathbf{q}(t) = \mathbf{q}_t, \\
& \qquad \mathbf{q}(s) \in Q, \quad \text{for all } s \in [t, t+T_p], \\
& \qquad \mathbf{u}(s) \in U, \quad \text{a.e. } s \in [t, t+T_c], && (7.2)\\
& \qquad \mathbf{u}(s) = k^{aux}(s, \mathbf{q}(s)), \quad \text{a.e. } s \in [t+T_c, t+T_p], \\
& \qquad \mathbf{q}(t+T_p) \in S.
\end{aligned}
$$

The domain of this optimization problem is the set of admissible processes, namely pairs $(\mathbf{q}, \mathbf{u})$ comprising a control function $\mathbf{u}$ and the corresponding state trajectory $\mathbf{x}$ which satisfy the constraints of $P(t, \mathbf{x}_t, T_c, T_p)$. A process $(\overline{\mathbf{q}}, \overline{\mathbf{u}})$ is said to solve $P(t, \mathbf{q}_t, T_c, T_p)$ if it minimizes (**??**) among all admissible processes.

Note that in the interval $[t+T_c, t+T_p]$ the control value is determined by $k^{aux}$ and therefore the optimization decisions are all carried out in the interval $[t, t+T_c]$.

As already said in Chapter 4, we call *design parameters* to those variables present in the open-loop optimal control problem that are not from the system model (i.e., variables we are able to choose); these comprise the control horizon $T_c$, the prediction horizon $T_p$, the running cost function $L$, the terminal cost function $W$, the auxiliary control law $k^{aux}$, and the terminal constraint set $S \subset \mathbb{R}^n$. The choice of these variables is important to obtain certain properties for the MPC strategy, such as stability, robustness, or performance (see e.g., [MRRS00, Fon01, FM03a] for a discussion on how to choose the design parameters).

The MPC algorithm performs according to a receding horizon strategy, as follows.

1. Measure the current state of the plant $\mathbf{q}^*(t_i)$;
2. Compute the open-loop optimal control $\overline{\mathbf{u}} : [t_i, t_i + T_c] \to \mathbb{R}^n$ solution to problem $P(t_i, \mathbf{q}^*(t_i), T_c, T_p)$;
3. Apply to the plant the control $\mathbf{u}^*(t) := \overline{\mathbf{u}}(t; t_i, \mathbf{q}^*(t_i))$ in the interval $[t_i, t_i + \delta[$ (the remaining control $\overline{\mathbf{u}}(t), t \geq t_i + \delta$ is discarded);
4. Repeat the procedure from 1. for the next sampling instant $t_{i+1}$ (the index $i$ is incremented by one unit).

The resultant control law $\mathbf{u}^*$ is a "sampling-feedback" control since during each sampling interval, the control $\mathbf{u}^*$ is dependent on the state $\mathbf{q}^*(t_i)$. More precisely, the resulting trajectory is given by

$$\mathbf{q}^*(t_0) = \mathbf{q}_{t_0}, \quad \dot{\mathbf{q}}^*(t) = f(t, \mathbf{q}^*(t), \mathbf{u}^*(t)) \quad t \geq t_0, \tag{7.3}$$

where

$$\mathbf{u}^*(t) := \overline{\mathbf{u}}(t; \lfloor t \rfloor_\pi, \mathbf{q}^*(\lfloor t \rfloor_\pi)) \quad t \geq t_0, \tag{7.4}$$

and the function $t \mapsto \lfloor t \rfloor_\pi$ gives the last sampling instant before $t$, that is

$$\lfloor t \rfloor_\pi := \max_i \{t_i \in \pi : t_i \leq t\}. \tag{7.5}$$

Similar sampled-data frameworks using continuous-time models and sampling the state of the plant at discrete instants of time were adopted in [CA98a, Fon01, Fon03, FIAF03, MS04] and are becoming the accepted framework for continuous-time MPC. It can be shown that with this framework it is possible to address — and guarantee stability, and robustness of the resultant closed-loop system — a very large class of systems, which are possibly nonlinear, time-varying, and nonholonomic.

## 7.3 The Two-Layer Control Scheme

As discussed, we propose a two-layer control scheme. The top layer, **the trajectory controller**, is applied to the group of vehicles as a whole. The trajectory controller is a nonlinear controller since most vehicles are nonholonomic systems and require a nonlinear, even discontinuous, feedback to stabilize them. The main ideas for the controller used in this section follow closely the results from section 6.2 of Chapter 6 [CF07].

The bottom layer, **the formation controller**, is computed and applied to each vehicle individually. The formation controller aims to compensate for small changes around a nominal trajectory maintaining the relative positions between vehicles. We argue that the formation control can be adequately carried out by a linear model predictive controller accommodating input and state constraints. This has the advantage that the control laws for each vehicle are simple piecewise affine feedback laws that can be pre-computed off-line and implemented in a distributed way in each vehicle. The main reference for this controller is [BMDP02].

## 7.4 The Vehicle Formation Models

### 7.4.1 Nonholonomic Vehicle Model For The Trajectory Controller

The framework developed here could easily be adapted to vehicles moving in 2D or 3D and having various dynamics. Nevertheless, we will explore a simple case of a differential-drive mobile robot moving on a plane, see Figure 2.4, represented by the following kinematic model:

$$\dot{x}(t) = (u_1(t) + u_2(t)) \cdot \cos\theta(t), \tag{7.6}$$

$$\dot{y}(t) = (u_1(t) + u_2(t)) \cdot \sin\theta(t), \tag{7.7}$$

$$\dot{\theta}(t) = (u_1(t) - u_2(t)), \tag{7.8}$$

with $\theta(t) \in [-\pi, \pi]$, and the controls $u_1(t), u_2(t) \in [-1, 1]$.

As we already know, the coordinates $(x, y)$ are the position in the plane of the midpoint of the axle connecting the rear wheels, and $\theta$ denotes the heading angle measured from the $x$-axis. The controls $u_1$ and $u_2$ are the angular velocity of the right and left wheels, respectively. If the same velocity is applied to both wheels, the robot moves along a straight line (maximum forward velocity when $u_1 = u_2 = u_{max} = 1$). The robot can turn by choosing $u_1 \neq u_2$ (when $u_1 = -u_2 = 1$ the robot turns counter-clockwise around the midpoint of the axle).

The velocity vector is always orthogonal to the wheel axis. This is the non-slip or nonholonomic constraint

$$(\dot{x}, \dot{y})^T (\sin\theta, -\cos\theta) = 0. \tag{7.9}$$

Therefore, the vehicle model is a nonholonomic system. It is completely controllable but instantaneously it cannot move in certain directions. It is known that this class of systems requires a nonlinear controller to stabilize it. Furthermore, the controller must allow discontinuous (or alternatively time-varying) feedback control laws. See [KM95] for a discussion on the control of nonholonomic systems, [Bro83, SS80] for the need to use discontinuous feedbacks, and [Fon03] for MPC of nonholonomic systems.

### 7.4.2 Linearized Vehicle Model for the Formation Controller

For the reasons explained above, to control the relative position between the vehicles in the formation, compensating for small deviations around a nominal trajectory, we might use linear control methods.

Consider that one of the vehicles is the reference vehicle. It might be the formation leader, or any of the vehicles in the formation, or we might even want to consider an additional nonexistent vehicle for modeling purposes. Consider that the formation moves at the nominal linear velocity $v_n$ and nominal angular velocity $w_n$ being the nominal velocities of the wheels. Let $v_l$ be the linear velocity added to the nominal linear velocity, and $v_w$ the angular velocity. Assume that $\theta$ is small, so that $\cos\theta \simeq 1$ and $\sin\theta \simeq \theta$. We thus have the simplified model

$$\dot{x}(t) = (v_n(t) + v_l(t)), \tag{7.10}$$

$$\dot{y}(t) = (v_n(t) + v_l(t))\theta(t), \tag{7.11}$$

$$\dot{\theta}(t) = v_w(t) + w_n(t). \tag{7.12}$$

We consider a linearized model for each of the differential drive mobile robots with the $z_1$ axis aligned with the velocity of the *reference vehicle*.

$$\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \\ \dot{z}_3 \end{bmatrix} = \begin{bmatrix} 0 & w_n & 0 \\ -w_n & 0 & v_n \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_l \\ v_w \end{bmatrix}, \tag{7.13}$$

which is smilar to the model obtained in Chapter 6 (6.19).

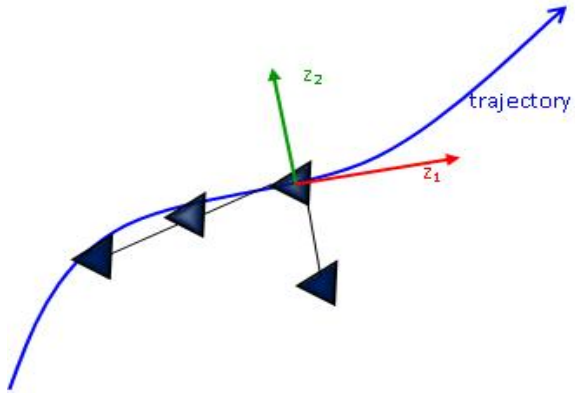


Figure 7.1: Reference frame for the linearized vehicle model aligned with the reference trajectory.

For computation purposes, it is convenient to use a discrete-time model here. The discrete-time model, converted using zero order hold with sample time $h$, is

$$\begin{bmatrix} \dot{z}_1(t+h) \\ \dot{z}_2(t+h) \\ \dot{z}_3(t+h) \end{bmatrix} = \begin{bmatrix} 0 & w_n h & 0 \\ -w_n h & 0 & v_n h \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} z_1(t) \\ z_2(t) \\ z_3(t) \end{bmatrix} + \begin{bmatrix} h & 0 \\ 0 & 0 \\ 0 & h \end{bmatrix} \begin{bmatrix} v_l(t) \\ v_w(t) \end{bmatrix} \tag{7.14}$$

which we will simply denote as

$$\dot{z}(t+h) = Az(t) + Bv(t). \tag{7.15}$$

### 7.4.3 Formation Connections Model

Consider $M$ vehicles and a *reference vehicle* (that might be fictitious) that follows exactly the trajectory predicted by the trajectory controller.

Consider that the $M+1$ vehicles are nodes of a directed graph $G = (V, E)$, which is a directed tree rooted at the reference vehicle. That is, all vertices are connected, there are no cycles, and all edges are directed in a path away from the root. Associate to each vertex $i$ is a triplet $z^i(t) = (z_1^i(t), z_2^i(t), z_3^i(t))$ with the relative position with respect to the reference vehicle at time $t$. Also, associate to each edge $(i, j) \in E$ a pair $\tilde{z}^{ij} = (\tilde{z}_1^{ij}, \tilde{z}_2^{ij})$ with the desired position in the plane of vehicle $j$ with respect to vehicle $i$. The desired relative positions are defined a priori and define the geometry of the formation that we want to achieve. For node $i$ we define its parent node $\mathcal{P}_i = j$ such that $(i, j) \in E$. See Figure 7.2.

The objective of the formation controller is to maintain the geometry of the formation, that is, the relative positions between the vehicles should be kept as close as possible to the desired relative positions. We define for each vehicle $i$ with parent $j$ a performance index that penalizes

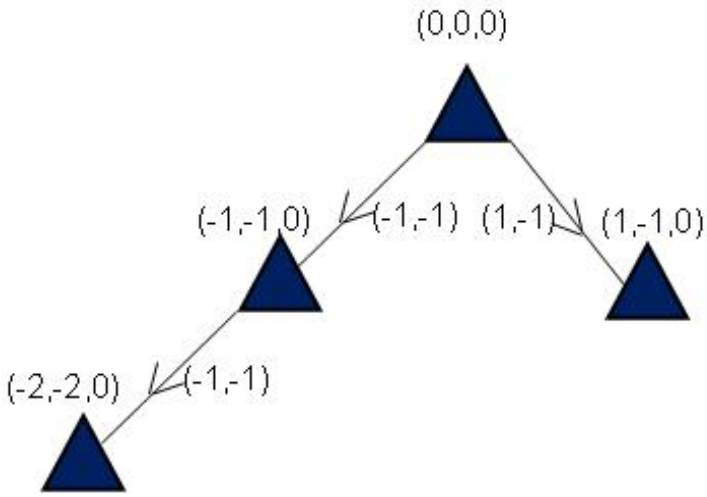


Figure 7.2: A tree modelling the formation connections

the difference between the desired and actual relative positions of the vehicles,

$$\begin{aligned} J^i =& \|(\tilde{z}_1^{ij}, \tilde{z}_2^{ij}, 0) - (z^j(N) - z^i(N))\|_P^2 + \\ &+ \sum_{t=1}^{N-1} \|(\tilde{z}_1^{ij}, \tilde{z}_2^{ij}, 0) - (z^j(t) - z^i(t))\|_Q^2 + \|(v(t), w(t))\|_R^2. \end{aligned} \tag{7.16}$$

## 7.5 The Controllers

### 7.5.1 Trajectory Controller

The controller used here follows closely the results in [CF07]. The aim is to devise a trajectory, and corresponding controls, for the formation as a whole.

The feedback controls used are $v$, $w$ as in Section 6.2 of Chapter 6 [CF07]. The feedbacks are obtained by solving, in a receding horizon strategy, the following optimal control problem

$$\begin{array}{ll} \text{Minimize} & \int_{t_i}^{t_i+T} L\left(\mathbf{q}(t), \mathbf{u}(t)\right) dt + W\left(\mathbf{q}\left(t_i+T\right)\right), \\ \text{subject to:} & \dot{\mathbf{q}}(t) = f\left(\mathbf{q}\left(t\right), \mathbf{u}\left(t\right)\right), \ \text{ a.e. } t \in [t_i, t_i+T], \\ & \mathbf{q}\left(t_i\right) = \mathbf{q}_{t_i}, \\ & \mathbf{u}\left(t\right) \in U\left(t\right), \ \text{ a.e. } t \in [t_i, t_i+T], \\ & \mathbf{q}\left(t_i+T\right) \in S. \end{array} \tag{7.17}$$

In this optimal control problem, the control horizon $T_c$ and the prediction horizon $T_p$ satisfy $T_c = N_c\delta$ and $T_p = N_p\delta$ with $N_c, N_P \in \mathbb{N}$ and $N_c \leq N_p$.

The guarantee of stability of the resulting closed loop system can be given by a choice of design parameters satisfying a sufficient stability condition. The following set of design parameters guarantees stability:

$$L(x, y, \theta) = x^2 + y^2 + \theta^2, \tag{7.18}$$

$$W(x, y, \theta) = \frac{1}{3}(r^3 + |\theta|^3) + r\theta^2 \quad \text{with} \quad r = \sqrt{x^2 + y^2}, \tag{7.19}$$

$$T_p = \pi, \tag{7.20}$$

$$\begin{aligned} S := \{(x, y, \theta) \in \mathbb{R}^2 \times [-\pi, \pi] : \phi_m(x, y) \leq \theta \leq \phi_M(x, y) \vee \\ \vee\ (x, y, \theta) \in \Theta \vee (x, y) = (0, 0)\}, \end{aligned} \tag{7.21}$$

where

$$\Theta = \left\{\mathbf{q} = (x, y, \theta) \in \mathbb{R}^2 \times [-\pi, \pi] : \|(x, y)\| \leq 0.05, \quad |\theta| \leq 0.1\right\}.$$

### 7.5.2 Formation controller

For each vehicle $i$, with parent $j$, compute the control solving the constrained linear quadratic optimal control problem:

Minimize over sequences $\{v_1, \ldots, v_{N-1}\}$,

$$J = \|(\tilde{z}_1^{ij}, \tilde{z}_2^{ij}, 0) - (z^j(N) - z(N))\|_P^2 + \sum_{t=1}^{N-1} \|(\tilde{z}_1^{ij}, \tilde{z}_2^{ij}, 0) - (z^j(t) - z(t))\|_Q^2 + \|v(t)\|_R^2, \tag{7.22}$$

subject to

$$\begin{aligned} z(t+h) &= Az(t) + Bv(t), & t = 0, ..., N-1, & \\ z(0) &= z_0, & & (7.23) \\ v_{\min} &\leq v(t) \leq v_{\max}, & t = 0, ..., N-1, & (7.24) \\ \left(\underline{z}_1^{ij}, \underline{z}_2^{ij}, 0\right) &\leq z(t) - z^j(t) \leq \left(\bar{z}_1^{ij}, \bar{z}_2^{ij}, 0\right), & t = 0, ..., N, & (7.25) \\ Dz(t) &\leq d, & t = 0, ..., N. & (7.26) \end{aligned}$$

Here, constraints (7.24) are limits on the inputs. Constraints (7.25) set a maximum and minimum distance to the parent vehicle, and inequalities (7.26) are general constraints to accommodate, for example, forbidden zones of the state-space.

The matrices involved in the performance index are chosen to satisfy $Q = Q^T \geq 0, R = R^T > 0$, and $P$ solving the Algebraic Ricatti Equation for discrete-time systems (see equation (3.41) in Section 3.5)

$$P = A^T PA - \left(A^T PB\right)\left(R + B^T PB\right)^{-1}\left(B^T PA\right) + Q.$$

This strategy has, for each vehicle and in the conditions stated, guaranteed exponential stability [BMDP02, MRRS00].

One of the major advantages of this controller is the fact that the feedback laws obtained from solving the linear quadratic regulator with constraints are continuous piecewise affine (PWA) functions [BMDP02], i.e., for a certain region of the state-space $\mathcal{R}_k$ (which is a polytope) the control law is affine.

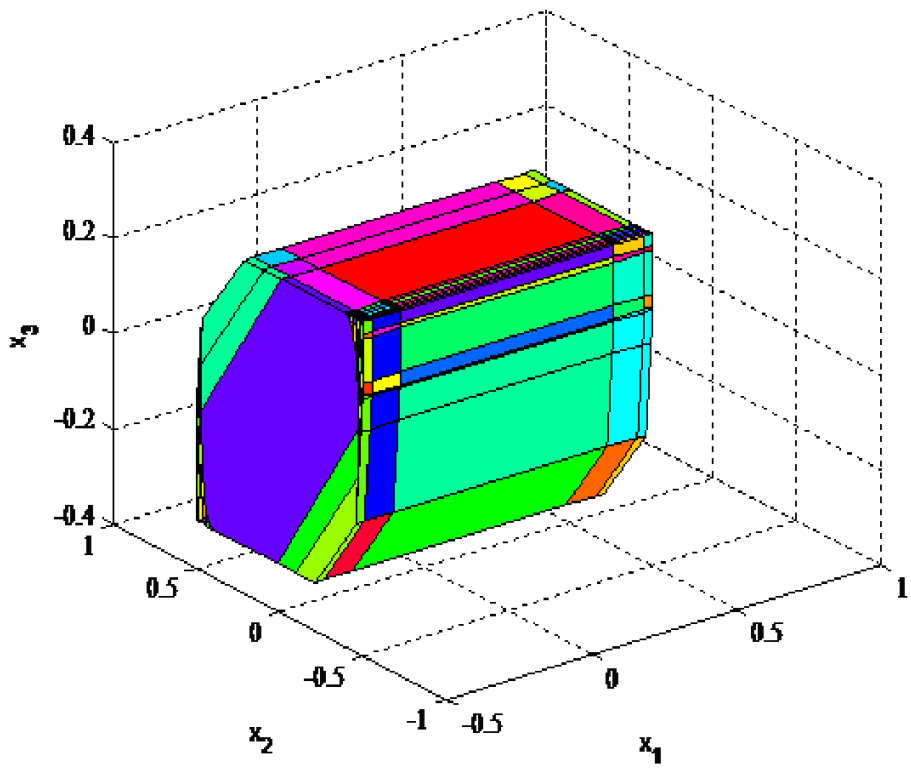


Figure 7.3: State-space regions for the reference vehicle PWA control law.

$$u(t) = F_k z(t) + G_k \qquad \text{if} \;\; z(t) \in \mathcal{R}_k. \tag{7.27}$$

In Figure 7.3 we can see the different state-space regions (195 polytopes) corresponding to different affine control laws for the reference vehicle.

This way, the parameters of the PWA feedback (matrices $F_k$ and $G_k$ for each region $\mathcal{R}_k$) can be determined explicitly a priori, off-line, by multi-parametric programming (e.g., using MPT - Multi Parametric Matlab Toolbox [KGBC06]). Each vehicle just has to store the parameters

of the PWA feedback function. No optimization or other complex computations are involved in real-time. Only lookup table operations are needed for each vehicle.

The stability of the whole formation is easy to establish. This is because, with the strategy above, the trajectory of each vehicle is exponentially stable with respect the desired relative position to its parent vehicle. As there is exactly one path from each vehicle to the reference vehicle, stability of any vehicle easily follows recursively.

If more general graphs are allowed, comprising not only trees but also admitting loops, the stability analysis is considerably more complex. For results on stability considering more general graphs see [DM06].

### 7.5.3 Integration of the two control layers

In each vehicle, the control given by the Trajectory Controller *plus* the control given by the Formation Controller are applied. Therefore, we have to consider control limits in each layer in such a way that the physical limits are respected. The Trajectory Controller can only use part of the limit (70% in our simulations) leaving the rest to the Formation Controller.

For the Trajectory Controller, the control law is computed centrally and then communicated to each vehicle. The frequency of update might be relatively low, and is dictated by the information from the outside (the formation) world of new obstacles in the trajectory, possibly identified by the sensors, that might alter the main path.

For the Formation Controller, the PWA feedback law is computed a priori, off-line, and implemented in each vehicle.

## 7.6 Notes at the end of Chapter

Formation control was inspired by the emergent self-organization observed in nature, like flocking birds and schooling fish. Each animal in a herd benefits by maximizing the chance of finding food and minimizing its encounters with predators. In addition, formation control allows for intelligent leaders and single agent planning while followers can focus on other tasks. The challenge here lies in designing a formation controller that is computationally simple.

In this Chapter, we proposed a two-layer scheme to control a set of vehicles moving in a formation. The first layer, the trajectory controller, is a nonlinear controller since most vehicles are nonholonomic systems and require a nonlinear, even discontinuous, feedback to stabilize them. The trajectory controller, a model predictive controller, computes control law and only a small set of parameters needs to be transmitted to each vehicle at each iteration. The second layer, the formation controller, aims to compensate for small changes around a nominal trajectory maintaining the relative positions between vehicles.

Model Predictive Control was shown to be an adequate tool for control of vehicles in a formation: it deals explicitly and effectively with the constraints that are an important problem feature; recent results on parameterized approaches to the optimal control problems allow addressing fast systems as well as allowing efficient implementations.

We argue that the formation control can be, in most cases, adequately carried out by a linear model predictive controller accommodating input and state constraints. This has the advantage that the control laws for each vehicle are simple piecewise affine feedback laws that can be pre-computed off-line and implemented in a distributed way in each vehicle.

Although several optimization problems have to be solved, the control strategy proposed results in a simple and efficient implementation where no optimization problem needs to be solved in real-time at each vehicle.

Most of the material in this chapter has been published in [FFC09], the main difference being that the trajectory controller used there was a robust controller doing the lines of [FM03b], while here we have chosen to use as the trajectory controller, the framework of [CF07], also described in Chapter 6.

# Chapter 8

# Optimal Reorganization of vehicle formations

## 8.1 Introduction

In this Chapter, we study the problem of switching the geometry of a formation of undistinguishable vehicles by minimizing some performance criterion. Given the initial positions and a set of final desirable positions, the questions addressed are:

1. Which vehicle should go to a specific final position?
2. How to avoid collision between the vehicles?
3. Which should be the traveling velocities of each vehicle between the initial and final positions?

Each vehicle can also modify its path by changing its curvature (for example to avoid obstacles). The performance criterion used in the example explored is to minimize the maximum traveling time, that is, we seek the allocation that minimizes the total formation switching time, but the method developed - based on dynamic programming - is sufficiently general to accommodate many different criteria.

The specific problem of switching the geometry of a formation arises in many cooperative vehicles missions, due to the need to adapt to environmental changes or to adapt to new tasks. An example of the first type is when a formation has to go through a narrow passage, or deviate from obstacles, and must reconfigure to a new geometry (see Figure 8.1). Examples of adaption to new tasks arise in robot soccer teams: when a team is in an attack formation and looses the ball, it should switch to a defense formation more appropriate to the new task, (see [LLCF09]).

Another example is in the detection and containment of a chemical spillage, the geometry of the formation for the initial task of surveillance, should change after detection occurs, switching to a formation more appropriate to determine the perimeter of the spill.

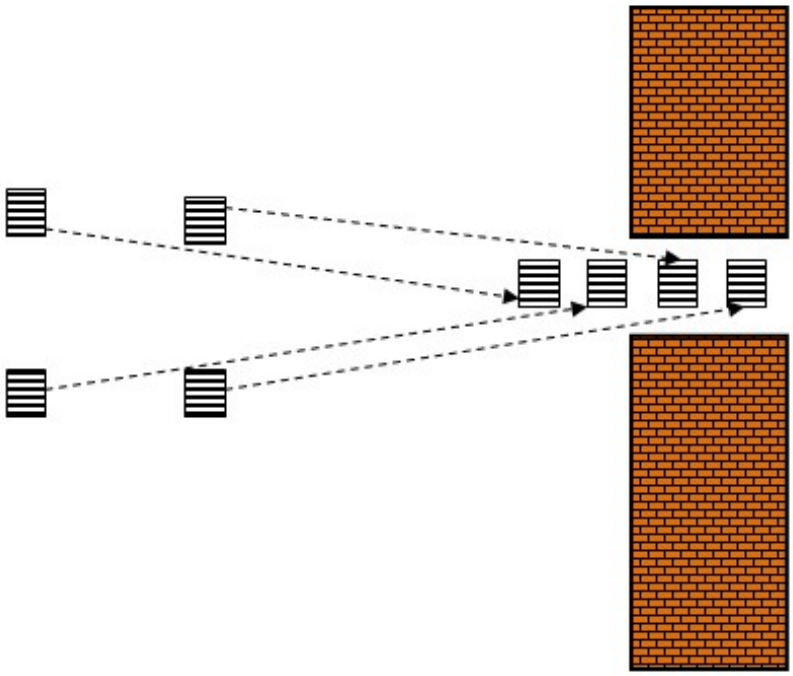

Figure 8.1: Reconfiguration of a formation to avoid obstacles.

Research in coordination and control of teams of several vehicles (that may be robots, ground, air or underwater vehicles) has been growing fast in the past few years. Application areas include unmanned aerial vehicles (UAVs) [BPA93, WCS96], autonomous underwater vehicles (AUVs) [SHL01], automated highway systems (AHSs) [Ben91, SH99] and mobile robotics [Yam99, YAB01]. While each of these application areas poses its own unique challenges, several common threads can be found. In most cases, the vehicles are coupled through the task they are trying to accomplish, but are otherwise dynamically decoupled, meaning that the motion of one does not directly affect the others. For a survey in cooperative control of multiple vehicles systems see, for example, the work by Murray [Mur07]. Regarding research on the optimal formation switching problem, it is not abundant, although it has been addressed by some authors. Desai et al., in [DOK01], model mobile robots formation as a graph. The authors use the so-called "control graphs" to represent the possible solutions for formation switching. In this method, for a graph having $n$ vertices there are $n!\,(n-1)!/2^{n-1}$ control graphs, and switching can only happen between predefined formations. The authors do not address collision or velocity issues. Hu and Sastry [HS01] study the problems of optimal collision avoidance and optimal formation switching for multiple vehicles on a Riemannian manifold. However, no choice of vehicle traveling velocity is considered. It is assumed that the underlying manifold

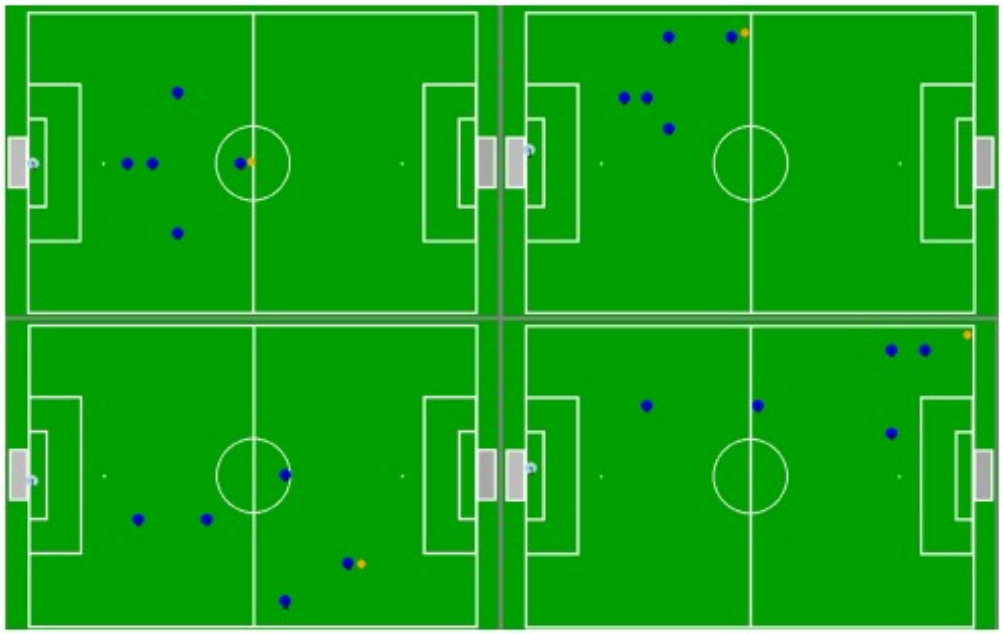

Figure 8.2: Different formation of soccer robots used for different ball position (from [LLCF09]).

admits a group of isometries, with respect to which the Lagrangian function is invariant. A reduction method is used to derive optimality conditions for the solutions. In [Yam04] Yamagishi describes a decentralized controller for the reactive formation switching of a team of autonomous mobile robots. The focus is on how a structured formation of vehicles can reorganize into a non-rigid formation based on changes in the environment. The controller utilizes nearest-neighbor artificial potentials (social rules) for collision-free formation maintenance and environmental changes act as a stimulus for switching between formations. A similar problem, where a set of vehicles must perform a fixed number of different tasks on a set of targets, has been addressed by several authors. The methods developed include exhaustive enumeration (see Rasmussen et al. [RSM+04]), branch-and-bound (see Rasmussen and Shima [RS06]), network models (see Schumacher et al. [SCR02, SCR03]), and dynamic programming (see Jin et al. [JSS06]). None of these works address velocity issues.

A problem of formation switching has also been addressed in [FF08b] and [FF10] using dynamic programming. The possible use of different velocities for each vehicle was addressed in [FFC11]. But the possibility of slowing down some of the vehicles might, as we will show in an example, achieve better solutions while avoiding collision between vehicles. Here we use a dynamic programming approach to solve the problem of formation switching with collision avoidance and vehicles velocities selection, that is the problem of deciding which vehicle moves to which place in the next formation guaranteeing that at any time the distance between any two of them is at least some predefined value. In addition, each vehicle can also explore the

possibility of modifying its velocity to avoid collision. Moreover, each vehicle can also modify its path by changing its curvature The formation switching performance is given by the time required for all vehicles to reach their new position, which is given by the maximum traveling time amongst individual vehicle traveling times. Since we want to minimize the time required for all vehicles to reach their new position, we have to solve a minmax problem. However, our methodology can be used with any separable performance function. The problem addressed here should be seen as a component of a framework for multivehicle coordination, incorporating also the trajectory control component, that allows to maintain or change formation while following a specified path in order to perform cooperative tasks.

This Chapter is organized as follows. In the next section, the problem of optimal reorganization of vehicle formations with collision avoidance is described and formally defined. In Section 3, a dynamic programming formulation of the problem is given and discussed. In Section 4, we discuss computational implementation issues of the dynamic programming algorithm, namely an efficient implementation of the main recursion as well as efficient data representations. A detailed description of the algorithms is also provided. Next, an example is reported to show the solution modifications when using velocities selection and collision avoidance. Some conclusions are drawn in the final section.

## 8.2 The Problem

In our problem a team of $N$ identical vehicles has to switch from their current formation to some other formation (i.e., vehicles have a specific goal configuration not related to the positions of the others), possibly unstructured, with collision avoidance. To address collision avoidance, we impose that the trajectories of the vehicles must satisfy the separation constraint that at any time the distance between any two of them is at least $\varepsilon$, for some positive $\varepsilon$, and to avoid obstacles, each vehicle can also modify its path by changing its curvature. The optimal (joint) trajectories are the ones that minimize the maximum trajectory time of individual vehicles.

Our approach can be used either centralized or decentralized, depending on the vehicles capabilities. In the latter case, all the vehicles would have to run the algorithm, which outputs an optimal solution, always the same if many exist, since the proposed method is deterministic.

Regarding the new formation, it can be either a pre-specified formation or a formation to be defined according to the information collected by the vehicles. In both cases, we do a pre-processing analysis that allows us to come up with the desired locations for the next formation.

This problem can be restated as the problem of allocating to each new position exactly one of the vehicles, located in the old positions, and determine each vehicle velocity. From all the possible solutions we are only interested in the ones where vehicle and obstacle collision is prevented. Among these, we want to find one that minimizes the time required for all vehicles to move to the target positions, that is an allocation which has the least maximum individual vehicle traveling time.

To formally define the problem, consider a set of $N$ vehicles moving in a space $\mathbb{R}^d$, so that at time $t$, vehicle $i$ has position $\mathbf{q}_i(t)$ in $\mathbb{R}^d$ (we will refer to $\mathbf{q}_i(t) = (x_i(t), y_i(t))$ when our space is the plane $\mathbb{R}^2$). The position of all vehicles is defined by the $N$-tuple

$$Q(t) = [\mathbf{q}_i(t)]_{i=1}^N$$

in $\mathbb{R}^{d\times N}$. We assume that each vehicle is holonomic and that we are able to choose its velocity, so that its kinematic model is a simple integrator

$$\dot{\mathbf{q}}_i(t) = \overline{v}_i(t) \quad \text{a.e.} \quad t \in \mathbb{R}^+.$$

The initial positions at time $t = 0$ are known and given by

$$A = [a_i]_{i=1}^N = Q(0).$$

Suppose a set of $M$ (with $M \geq N$) final positions in $\mathbb{R}^d$ is specified as

$$F = \{f_1, f_2, ..., f_M\}.$$

The problem is to find an assignment between the $N$ vehicles and $N$ of the $M$ final positions in $F$. That is, we want to find a $N$-tuple $B = [b_i]_{i=1}^N$ of different elements of $F$, such that at some time $T > 0$,

$$Q(T) = B$$

and all $b_i \in F$, with $b_i \neq b_k$.

There are

$$\begin{pmatrix} M \\ N \end{pmatrix} \cdot N!$$

such $N$-tuples (the permutations of a set of $N$ elements chosen from a set of $M$ elements) and we want to find a procedure to choose an $N$-tuple minimizing a certain criterion that is more efficient than total enumeration.

The criterion to be minimized can be very general since the procedure developed is based on dynamic programming which is able to deal with general cost functions.

Examples can be:

- minimizing the total distance traveled by the vehicles

$$Minimize \sum_{i=1}^{N} \|b_i - a_i\|,$$

- or, the total traveling time

$$Minimize \sum_{i=1}^{N} \|b_i - a_i\| / \|\overline{v}_i\|,$$

- or, the maximum traveling time

$$Minimize \qquad \max_{i=1,\ldots,N} \|b_i - a_i\| \,/\, \|\overline{v}_i\| \,,$$

We are also interested in selecting the traveling velocities of each vehicle. Assuming constant velocities, these are given by

$$\overline{v}_i\,(t) = \overline{v}_i = v_i \frac{b_i - a_i}{\|b_i - a_i\|},$$

where the constant speeds are selected from a discrete set

$$\Upsilon = \{V_{\min}, ..., V_{\max}\}\,.$$

Moreover, we are also interested in avoiding collision between vehicles. We say that two vehicles $i, k$ (with $i \neq k$), do not collide if their trajectories maintain a certain distance apart, at least $\varepsilon$, at all times. The non-collision conditions is

$$\|\mathbf{q}_i(t) - \mathbf{q}_k(t)\| \geq \varepsilon, \qquad \forall t \in [0, T]\,, \tag{8.1}$$

where the trajectory, in the linear case, is given by

$$\mathbf{q}_i(t) = a_i + \overline{v}_i(t)\ t, \qquad t \in [0, T]\,.$$

We can then define a logic-valued function $c$ (see Figure 8.3) as

$$c\,(a_i, v_i, b_i, a_k, v_k, b_k) = \begin{cases} 1 & \text{if collision between } i \text{ and } k \text{ ocurs,} \\ 0 & \text{otherwise.} \end{cases}$$

With these considerations, the problem (in the case of minimizing the maximum traveling time) can be formulated as follows

$$\begin{array}{lll} \min\limits_{b_1,\ldots,b_N,v_1,\ldots,v_N} & \max\limits_{i=1,\ldots,N} & \|b_i - a_i\| \ / \ v_i, \\ \text{Subject to} & & \\ & b_i \in F, & \forall i, \\ & b_i \neq b_k, & \forall i, k \ \text{ with } \ i \neq k, \\ & v_i \in \Upsilon, & \forall i, \\ & c\,(a_i, v_i, b_i, a_k, v_k, b_k) = 0, & \forall i, k \ \text{ with } \ i \neq k. \end{array}$$

Instead of using the set $F$ of $d$-tuples, we can define a set $J = \{1, 2, ..., M\}$ of indexes to such $d$-tuples, and also a set $I = \{1, 2, ...N\}$ of indexes to the vehicles. Let $j_i$ in $J$ be the target position for vehicle $i$, that is, $b_i = f_{j_i}$. Define also the distances $d_{ij} = \|f_j - a_i\|$ which can be pre-computed for all $i \in I$ and $j \in J$. Redefining, without changing the notation, the function $c$ to take as arguments the indexes to the vehicle positions instead of the positions, i.e.,

$$c\,(a_i, v_i, f_{j_i}, a_k, v_k, f_{j_k})$$

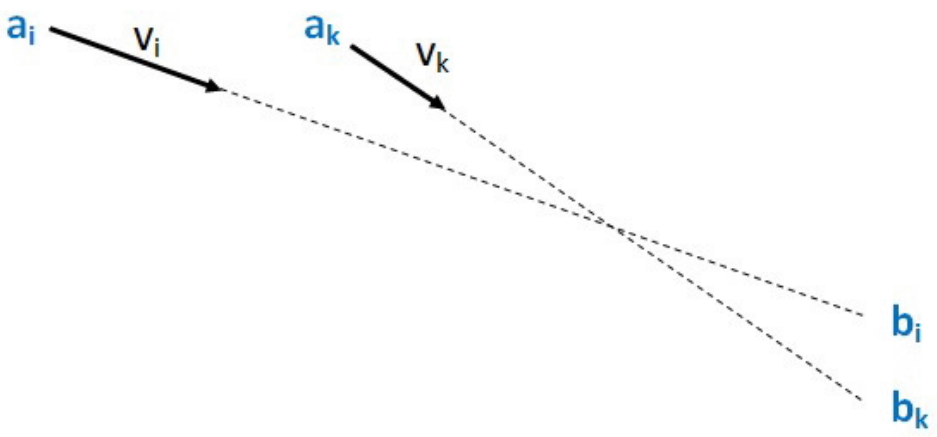


Figure 8.3: Linear trajectories of two vehicles.

is simply represented as

$$c\left(a_i, v_i, j_i, a_k, v_k, j_k\right).$$

The problem can be reformulated into the form

$$\begin{array}{lll} \min\limits_{j_1,\ldots,j_N,v_1,\ldots,v_N} & \max\limits_{i=1,\ldots,N} & d_{ij} \ / \ v_i, \\ \text{Subject to} & & \\ & j_i \in J, & \forall i \in I, \\ & j_i \neq j_k, & \forall i,k \in I \ \text{ with } \ i \neq k, \\ & v_i \in \Upsilon, & \forall i \in I, \\ & c\left(a_i, v_i, j_i, a_k, v_k, j_k\right) = 0, & \forall i,k \in I \ \text{ with } \ i \neq k. \end{array}$$

We are finally in position to consider the full problem with obstacle avoidance. Consider that the vehicle $a$ is moving in space $\mathbb{R}^2$ and travels at velocity $v_a$ to position $b$ that is reached in time $T_a$. Suppose that there is an obstacle, and if the vehicle $a$ travels straight ahead collides with the obstacle, then, the vehicle $a$ should circumvents the obstacle on the top or on the botton (see Figure 8.4).

Now, the problem can be formulated in a similar way where $d_{ij}\left(p_i\right)$ is the length of the path from $i$ to $j$ using path $p_i$ and $c$ is also dependent on the path $p_i$.

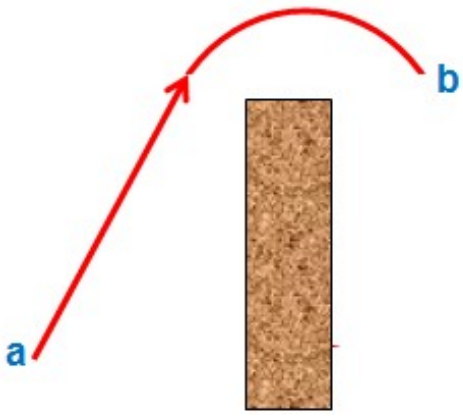


Figure 8.4: Vehicle $a$ circumvents the obstacle on top.

$$
\begin{array}{lll}
\min\limits_{j_1,\ldots,j_N,v_1,\ldots,v_N,p_1,\ldots,p_N} & \max\limits_{i=1,\ldots,N} & d_{ij}\left(p_i\right) \ / \ v_i \\
\text{Subject to} & & \\
 & j_i \in J, & \forall i \in I, \\
 & j_i \neq j_k, & \forall i,k \in I \ \text{ with } \ i \neq k, \\
 & v_i \in \Upsilon, & \forall i \in I, \\
 & p_i \in P_i\left(j\right), & \forall i \in I \\
 & c\left(a_i, v_i, j_i, a_k, v_k, j_k, p_i, p_k\right) = 0, & \forall i,k \in I \ \text{ with } \ i \neq k.
\end{array}
$$

where $P_i\left(j\right)$ is the set of all paths from $i$ to $j$.

## 8.3 Dynamic programming formulation

Dynamic Programming (DP) is an effective method to solve combinatorial problems of a sequential nature. It provides a framework for decomposing an optimization problem into a nested family of subproblems. This nested structure suggests a recursive approach for solving the original problem using the solution to some subproblems. The recursion expresses an intuitive *principle of optimality* [Bel57] for sequential decision processes, that is, once we have reached a particular state, a necessary condition for optimality is that the remaining decisions must be chosen optimally with respect to that state.

### 8.3.1 Derivation of the dynamic programming recursion: the simplest problem

We start by deriving a DP formulation for a simplified version of problem: where collision is not considered and different velocities are not selected. The collision avoidance, the selection of velocities for each vehicle, and the changing of the curvature path of each vehicle are introduced later.

Consider that there are $N$ vehicles $i = 1, 2, ..., N$ to be relocated from known initial location coordinates to a target locations indexed by set $J$. We want to allocate exactly one of the vehicles to each position in the new formation. In our model a stage $i$ contains all states $S$ such that $|S| \geq i$, meaning that $i$ vehicles have been allocated to the targets in $S$. The DP model has $N$ stages, with a transition occurring from a stage $i-1$ to a stage $i$, when a decision is made about the allocation of vehicle $i$.

Define $f(i, S)$ to be the value of the best allocation of vehicles $1, 2, ..., i$ to $i$ targets in set $S$, that is, the allocation requiring the least maximum time the vehicles take to go to their new positions. Such value is found by determining the least maximum vehicle traveling time between its current position and its target position. For each vehicle $i$, the traveling time to the target position $j$ is given by

$$d_{ij} \ / \ v_i.$$

By the previous definition, the minimum traveling time of the $i-1$ vehicles to the target positions in set $S \setminus \{j\}$ is given by $f(i-1, S \setminus \{j\})$. From the above, the minimum traveling time of all $i$ vehicles to the target positions in $S$ they are assigned to, given that vehicle $i$ travels at velocity $v_i$, without vehicle collisions, is obtained by examining all possible target locations $j \in S$ (see Figure 8.5).The *dynamic programming recursion* is then defined as

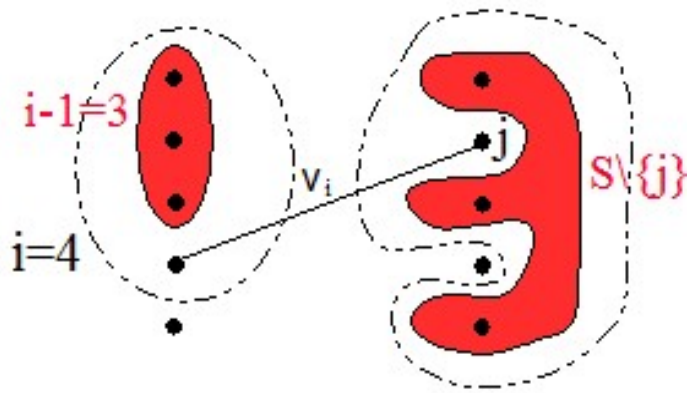


Figure 8.5: Dynamic Programming Recursion with N = 5 and stage i = 4.

$$f(i, S) = \min \{d_{ij} \ / \ v_i \ \vee \ f(i-1, S \setminus \{j\})\}. \tag{8.2}$$

where $X \vee Y$ denotes the maximum between $X$ and $Y$.

The *initial conditions* for the above recursion are provided by

$$f(1, S) = \min\{d_{1j} \ / \ v_1\}, \qquad \forall S \subseteq J \tag{8.3}$$

and all other states are initialized as not yet computed.

Hence, the optimal value for the performance measure, that is, the minimum traveling time needed for all $N$ vehicles to assume their new positions in $J$, is given by

$$f(N, J). \tag{8.4}$$

### 8.3.2 Considering collision avoidance and velocities selection

Recall function $c$ for which $c(i, v_i, j, a, v_a, b)$ takes value 1 if there is collision between pair of vehicles $i$ and $a$ traveling to positions $j$ and $b$ with velocities $v_i$ and $v_a$, respectively, and takes value 0 otherwise. To analyze if the vehicle traveling through a newly defined trajectory collides with any vehicle traveling through previously determined trajectories, we define a recursive function. This function checks the satisfaction of the collision condition, given by equation (8.1), in turn, between the vehicle which had the trajectory defined last and each of the vehicles for which trajectory decisions have already been made.

So, if the vehicles $i$ and $a$ go straight ahead, i.e., the path for both will be straight ahead the collision between these two vehicles occurs if the following condition is satisfied:

$$\left\| (x_i, y_i) + v_i \frac{(x_j, y_j) - (x_i, y_i)}{\|(x_j, y_j) - (x_i, y_i)\|} t - \left[ (x_a, y_a) + v_a \frac{(x_b, y_b) - (x_a, y_a)}{\|(x_b, y_b) - (x_a, y_a)\|} t \right] \right\| < \varepsilon$$

for some $t \in [0, \min\{T_i, T_a\}]$ (see [FF10]).

We note that by trajectory we understand not only the path between the initial and final positions but also a timing law and an implicitly defined velocity.

Consider that we are in state $(i, S)$ and that we are assigning vehicle $i$ to target $j$. Further let $v_{i-1}$ be the traveling velocity for vehicle $i-1$. Since we are solving state $(i, S)$ we need state $(i-1, S \backslash \{j\})$, which has already been computed (if this is not the case, then we must compute it first). In order to find out if this new assignment is possible, we need to check if at any point in time vehicle $i$, traveling with velocity $v_i$, will collide with any of the vehicles $1, 2, ..., i-1$ for which we have already determined the target assignment and traveling velocities.

Let us define a recursive function (see Figure 8.6)

$$C(i, v_i, j, k, V, S)$$

that assumes the value one if a collision occurs between vehicle $i$ traveling with velocity $v_i$ to $j$ and any of the vehicles $1, 2, ...k$, with $k < i$, traveling to their targets, in set $S$, with their respective velocities $V = [v_1 \ v_2 \ ... \ v_k]$ and assumes the value zero if no such collisions occurs. This function works in the following way (see Figure 8.7):

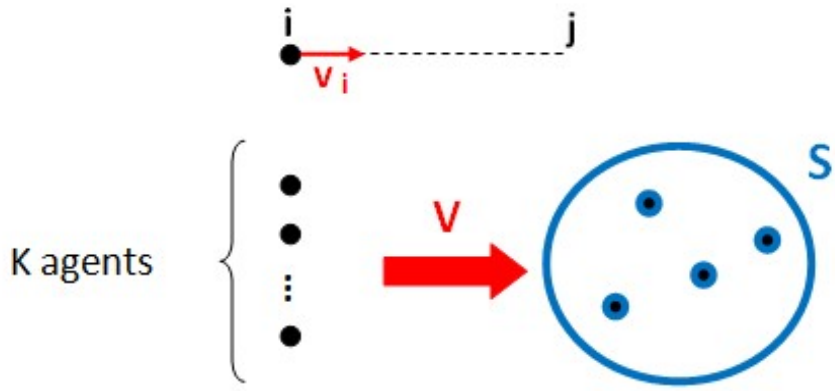


Figure 8.6: Collision between a vehicle and a set of vehicles.

1. first it verifies $c\left(i, v_i, j, k, v_k, Be_j\right)$, that is, it verifies if there is collision between trajectory $i \longrightarrow j$ at velocity $v_i$ and trajectory $k \longrightarrow Be_j$ at velocity $v_k$, where $Be_j$ is the optimal target for vehicle $k$ when targets in set $S \backslash \{j\}$ are available for vehicles $1, 2, ..., k$. If this is the case it returns the value 1;

2. Otherwise, if they do not collide, it verifies if trajectory $i \longrightarrow j$ at velocity $v_i$ collides with any of the remaining vehicles. That is, it calls the collision function $C\left(i, v_i, j, k-1, V', S'\right)$, where $S' = S \backslash \{Be_j\}$ and $V = [V' \ v_k]$.

The collision recursion is therefore written as:

$$C\left(i, v_i, j, k, V, S\right) = \left\{c\left(i, v_i, j, k, v_k, Be_j\right) \ \vee \ C\left(i, v_i, j, k-1, V', S'\right)\right\} \tag{8.5}$$

where

$$Be_j = Best_j\left(k, V', S'\right),$$

and

$$V = [V' \ v_k], \qquad S' = S \backslash \{j\}.$$

The initial conditions for recursion (8.5) are provided by

$$C\left(i, v_i, j, 1, v_1, \{k\}\right) = \left\{c\left(i, v_i, j, 1, v_1, k\right)\right\}$$

$\forall i \in I; \ \forall j, k \in J$ with $j \neq k; \ \forall v_i, v_1 \in \Upsilon$. All other states are initialized as not yet computed.

The dynamic programming recursion for the minimal time switching problem with collision avoidance and velocities selection is then

$$f\left(i, V, S\right) = \min\left\{d\left(i, j\right) \ / \ v_i \ \vee \ f(i-1, V', S') \ \vee \ \widetilde{M} * C\left(i, v_i, j, i-1, V', S'\right)\right\}, \tag{8.6}$$

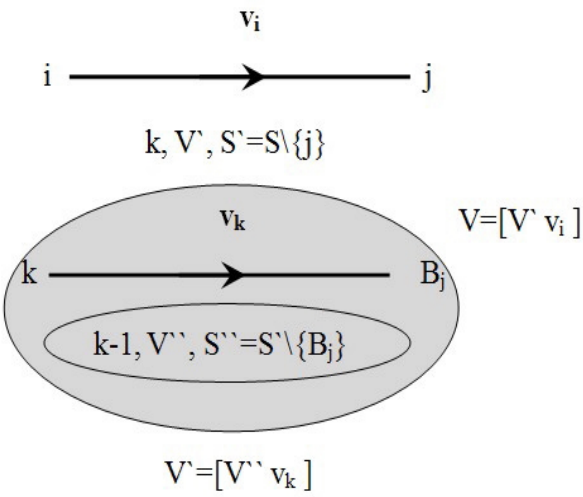


Figure 8.7: Collision recursion.

where $V = [V' \ v_i]$, $S' = S \backslash \{j\}$ and $C$ is the collision function where $\widetilde{M}$ is a sufficiently large number so that any solution with collision is eliminated.

The initial conditions are given by

$$f(1, v_1, \{j\}) = \min \{d(1, j) \ / \ v_1\}, \quad \forall j \in J \ \text{ and } \ \forall v_1 \in \Upsilon.$$

All other states being initialized as not computed.

To determine the optimal value for our problem we have compute

$$\min_{\text{all } N\text{-tuples } V} \quad f(N, V, J).$$

### 8.3.3 Considering obstacles

Consider the following pair of vehicles moving in space $\mathbb{R}^2$: vehicle $i$ which travels at velocity $v_i$ to position $j$ that is reached in time $T_i$ and the vehicle $a$ which travels at velocity $v_a$ to position $b$ that is reached in time $T_a$. Collision between these two vehicles occurs if the following condition is satisfied (see equation (8.1)):

$$\|q_i(t) - q_a(t)\| < \varepsilon \tag{8.7}$$

where $q_i(t) = (x_i(t), y_i(t))$ and $q_a(t) = (x_a(t), y_a(t))$.

Assuming now that there is an obstacle, and if the vehicle $a$ travels straight ahead collides with the obstacle (see Figure 8.8). What should the vehicle $a$ do?

Vehicle $a$ travels straight ahead and intersects an obstacle.

Suppose that the obstacle is the rectangle $[Q_1Q_2Q_3Q_4]$ (see Figure 8.9), if not we can always put the obstacle inside of one rectangle. Our vehicle $a$ is in the position $A = (x_a, y_a)$ and the

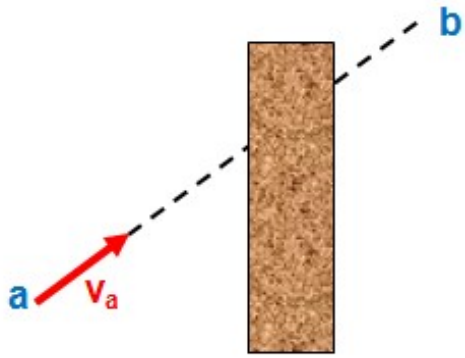


Figure 8.8: Vehicle $a$ travels straight ahead and intersects an obstacle.

goal is the position $B = (x_b, y_b)$ which is reached in time $T_a$. If the straight line $[AB]$ does not intersect the obstacle then the path will be straight ahead. Otherwise, we take the midpoint $P$ of the straight line resulting from the intersection of the line $[AB]$ with the obstacle. Calculate the point $A_1 = (x_1, y_1)$, so that the point $P$ is the midpoint of the straight line $[A_1B]$.

Let $Q_0 \in \{Q_1, Q_2, Q_3, Q_4\}$. Considering the minimum between $d_1$ and $d_2$, where $d_1$ is the maximum distance between $P$ and $Q_1$ and $P$ and $Q_2$ and $d_2$ is the maximum distance between $P$ and $Q_3$ and $P$ and $Q_4$. The point $Q_0$ is the vertex of the rectangle that checks this minimum. If point $Q_0$ is determined by taking into account the distance $d_1$ then the vehicle circumvents the obstacle on top, otherwise, if $Q_0$ is determined by taking into account the distance $d_2$ then the obstacle is outlined below (see Figure 8.9-1).

Let $\xi \in \mathbb{R}^+$ and consider $Q = (x_q, y_q) = Q_0 + \xi \overrightarrow{PQ_0}$. Consider the arc of the circumference that passes through the points $A_1$, $B$ and $Q$. The center of this circumference is determined by the intersection of the line bisection of the segment $[A_1Q]$ and the line bisection of the segment $[BQ]$. Let $C = (c_1, c_2)$ be the center of this circumference. The radius is $r = \overline{A_1C}$ (see Figure 8.9-2).

Defining $\alpha = \arctan\left(\frac{y_q - y_c}{x_q - x_c}\right)$ and $\theta = \arccos\left(\frac{d}{r}\right)$ (see Figure 8.10), where $d$ is the distance between the midpoint of the line segment $[QB]$ and $C$. After time $\tau$ the vehicle travels from $\alpha$ to $\alpha - 2\theta$ at speed $v_a$ taking $T_a - \tau = \frac{2\theta r}{v_a}$ seconds.

So the vehicle follows a straight line from point $A$ to point $Q$ which is reached in time $\tau$, then it follows an arc of circunference from point $Q$ to point $B$ defined as follows (see Figure 8.9-3):

$$\begin{cases} x(t) = c_1 + r\cos\left(\alpha - 2\frac{t-\tau}{T_a-\tau}\theta\right), \\ y(t) = c_2 + r\sin\left(\alpha - 2\frac{t-\tau}{T_a-\tau}\theta\right), \end{cases}, \qquad t \in [\tau, T_a].$$

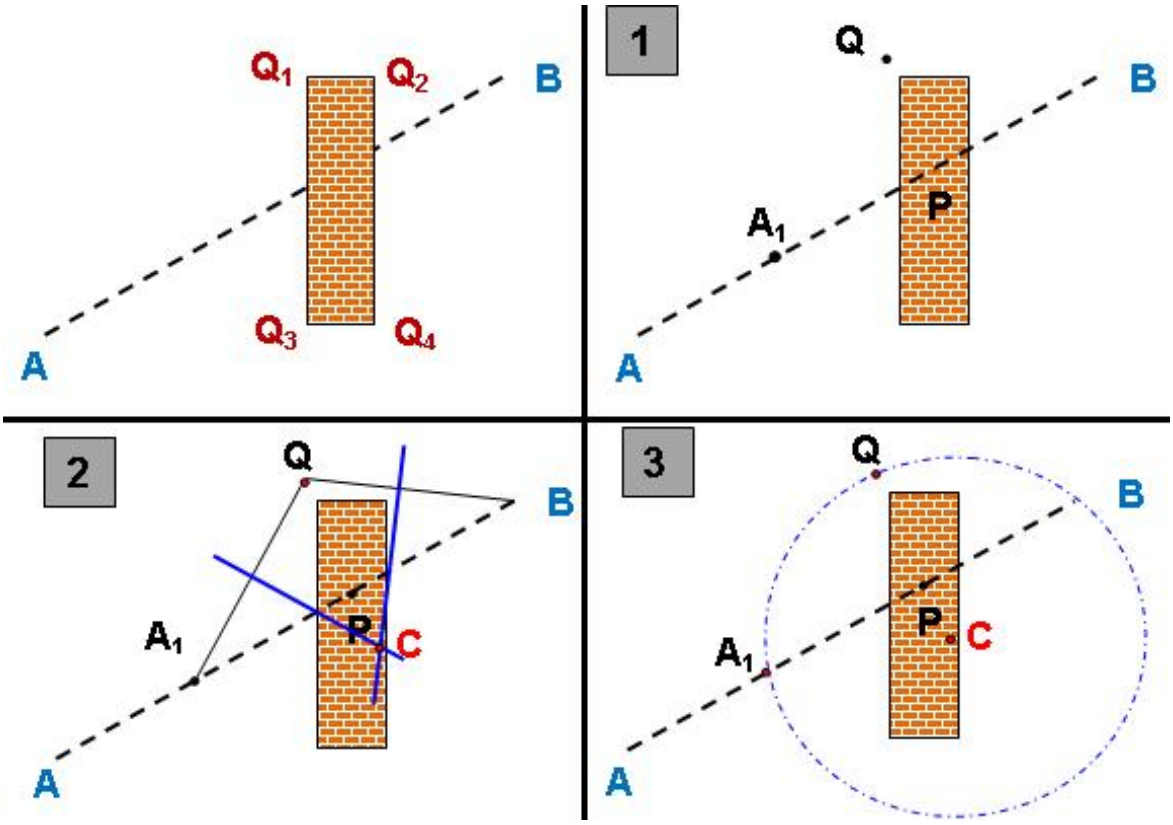


Figure 8.9: Trajectory of the vehicle $a$ from the position $A$ to position $B$ if there is an obstacle in the way.

The trajectory of the vehicle $a$ is, if $\tau \neq T_a$, then $q_a(t) = (x_a(t), y_a(t))$ where:

$$q_a(t) = \begin{cases} (x_a(0) + v_a t, y_a(0) + v_a t), & \text{if} \quad t \in [0, \tau], \\ \left(c_1 + r\cos\left(\alpha - 2\frac{t-\tau}{T_a-\tau}\theta\right), c_2 + r\sin\left(\alpha - 2\frac{t-\tau}{T_a-\tau}\theta\right)\right), & \text{if} \quad t \in [\tau, T_a]. \end{cases} \tag{8.8}$$

otherwise

$$q_a(t) = (x_a(0) + v_a t, y_a(0) + v_a t) \quad \text{for} \quad t \in [0, T_a]. \tag{8.9}$$

The logic-valued collision funtion $C$ that checks whether a pair of vehicles colides is now redefined using condition (8.7) with the trajectories given by (8.8) or (8.9). Thus,

$$c(i, v_i, j, a, v_a, b, p_i, p_a) = 0$$

if

$$\|q_i(t) - q_a(t)\| < \varepsilon,$$

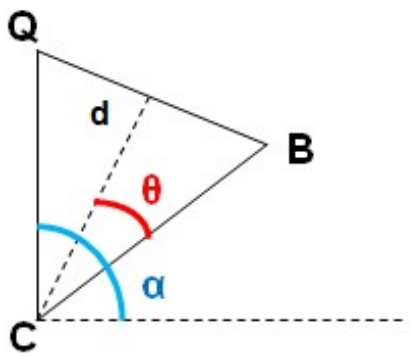


Figure 8.10: The angles $\theta$ and $\alpha$.

where $q_i(t)$ an $q_a(t)$ are defined by equations (8.8), that define the position along the path $p_i$ (or $p_a$) for all times.

The objective function is now dependent on the length along the path $p_i$, that is $d_{ij}(p_i)$.

The length of the arc of circunference from point $Q$ to point $B$ is:

$$\begin{aligned} & \int_{\tau}^{T_a} \sqrt{\left[-\frac{2r\theta}{T_a-\tau}\sin\left(\alpha-2\frac{t-\tau}{T_a-\tau}\theta\right)\right]^2+\left[\frac{2r\theta}{T_a-\tau}\cos\left(\alpha-2\frac{t-\tau}{T_a-\tau}\theta\right)\right]^2}\,dt \\ = & \int_{\tau}^{T_a} \frac{2r\theta}{T_a-\tau}\,dt = \frac{2r\theta}{T_a-\tau}(T_a-\tau) = 2r\theta. \end{aligned}$$

So,

$$d_{ij}(p_i) = \begin{cases} d_{ij} & \text{if the vehicle } a \text{ travels straight ahead} \\ \widetilde{d}_{ij} + 2r\theta & \text{if the vehicle } a \text{ intersects an obstacle.} \end{cases}$$

where $\widetilde{d}_{ij} = \left\|\overrightarrow{AQ}\right\|$.

## 8.4 Computational implementation

The DP procedure we have implemented exploits the recursive nature of the DP formulation by using a backward-forward procedure. Although a pure forward Dynamic Programming (DP) algorithm can be easily derived from the DP recursion, equations (8.2) and (8.6), such implementation would result in considerable waste of computational effort since, generally, complete computation of the state space is not required. Furthermore, since the computation of a state requires information contained in other states, rapid access to state information should be seek.

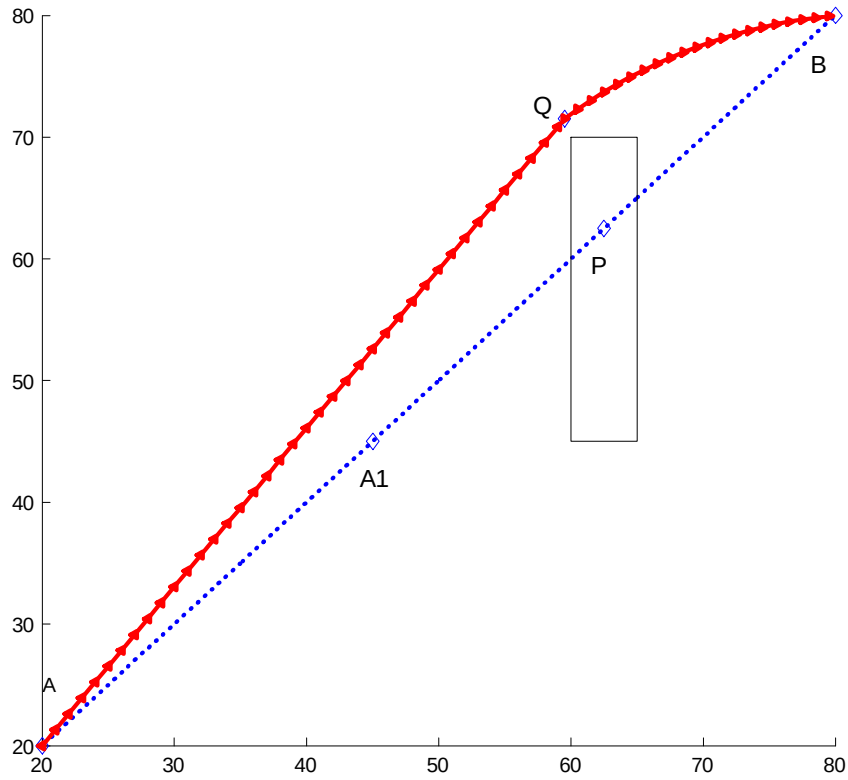


Figure 8.11: Trajectory of vehicle $a$.

The main advantage of the backward-forward procedure implemented is that the exploration of the state space graph, i.e., the solution space, is based upon the part of the graph which has already been explored. Thus, states which are not feasible for the problem are not computed, since only states which are needed for the computation of a solution are considered. The algorithm is dynamic as it detects the needs of the particular problem and behaves accordingly.

States at stage 1 are either nonexistent or initialized as given in equation (8.6). The DP recursion, equation (8.2), is then implemented in a backward-forward recursive way. It starts from the final states $(N, V, J)$ and while moving backward visits, without computing, possible states until a state already computed is reached. Initially, only states in stage 1, initialized by equation (8.6) are already computed. Then, the procedure is performed in reverse order, i.e., starting from the state last identified in the backward process, it goes forward through computed states until a state $(i, V', S')$ is found which has not yet been computed. At this point, again it goes backward until a computed state is reached. This procedure is repeated

until the final states $(N, V, J)$ for all $V$ are reached with a value that cannot be improved by any other alternative solution. From these we choose the minimum one. The main advantage of this backward-forward recursive algorithm is that only intermediate states needed are visited and from these only the feasible ones that may yield a better solution are computed.

As said before, due to the recursive nature of equation (8.2), state computation implies frequent access to other states. Recall that a state is represented by a number, a sequence, and a set. Therefore, sequence operations like adding or removing an element and set operations like searching, deletion and insertion of a set element must be performed efficiently. So, the question arises:

How to represent sets $S$ and sequences $V$ by integers in an efficient way?

The answer to these problems will be given in what follows, each set is represented by a binary string of lenght $n$ (*bit-set representation*) and we represent a sequence as a numeral with $n$ digits in the base $k$.

### 8.4.1 Sequence representation and operation

A sequence is a $n$-tuple with $k$ possible values for each element, where $n$ is the number of vehicles and $k$ the number of possible velocity values. Therefore, there are $k^n$ possible sequences to be represented. If sequences are represented by integers in the range $0 \sim k^n - 1$ then it is easy to implement sequence operations such as partitions. Thus, we represent a sequence as a numeral with $n$ digits in the base $k$. The partition of a sequence with $l$ digits that we are interested on is the one corresponding to the first $l - 1$ digits and the last digit. Such a partition can be obtained by performing the integer division in the base $k$ and taking the remainder of such division.

**Example 5** *Consider a sequence of length $n = 4$ with $k = 3$ possible values $v_0, v_1$, and $v_2$. This is represented by numeral with $n$ digits in the base $k$*

$$[v_1, v_0, v_2, v_1] \textit{ is represented by } 1\ 0\ 2\ 1\ _3 = 1 \cdot 3^3 + 0 \cdot 3^2 + 2 \cdot 3^1 + 1 \cdot 3^0 = 34$$

Partition of this sequence by the last element can be performed by integer division in the base $k$ and taking the remainder of such division,

$$V = 1\ 0\ 2\ 1\ _3 = 34$$

can be split into $[V'\ v_i]$ as follows:

$$V' = 1\ 0\ 2\ _3 = 1 \cdot 3^2 + 2 \cdot 3^0 = 11 = 34\ DIV\ 3$$

and

$$v_i = 1_3 = 1 = 34\ MOD\ 3.$$

### 8.4.2 Set representation and operation

A computationally efficient way of storing and operating sets is the bit-vector representation, also called the boolean array, whereby a *computer word* is used to keep the information related to the elements of the set. In this representation a universal set $U = \{1, 2, ..., n\}$ is considered. Any subset of $U$ can be represented by a binary string (a computer word) of length $n$ in which the $ith$ bit is set to 1 if $i$ is an element of the set, and set to 0 otherwise. So, there is a one-to-one correspondence between all possible subsets of $U$ (in total $2^n$) and all binary strings of length $n$. Since there is also a one-to-one correspondence between binary strings and integers, the sets can be efficiently stored and worked out simply as integer numbers. A major advantage of such implementation is that the set operations, *location, insertion or deletion* of a set element can be performed by directly addressing the appropriate bit. For a detailed discussion of this representation of sets see, for example, the book by Aho et al. [AHU83].

**Example 6** *Consider the Universal set* $U = \{1, 2, 3, 4\}$ *of* $n = 4$ *elements. This set and any of its subsets can be represented by a binary string of length* 4*, or equivalently its representation as an integer in the range* $0 \sim 15$.

$$U = \{1, 2, 3, 4\} \quad \textit{is represented by} \quad 1111_B = 15.$$

$$\textit{A subset } A = \{1, 3\} \quad \textit{is represented by} \quad 1010_B = 10.$$

### 8.4.3 Algorithms

The flow of the algorithm is managed by **Algorithm 1**, which starts by labeling all states (subproblems) as not yet computed, that is, it assigns to them a $\infty$ value. Then, it initializes states in stage 1, that is subproblems involving 1 vehicle, as given by equation (8.6). After that, it calls **Algorithm 2** with parameters $(N, V, J)$. **Algorithm 2**, that implements recursion (8.2), calls **Algorithm 3** to check for collisions every time it attempts to define one more vehicle-target allocation. This algorithm is used to find out whether the newly established allocation satisfies the collision regarding all previously defined allocations or not, feeding the result back to **Algorithm 2**. **Algorithm 4**, called after **Algorithm 2** has finished, also implements a recursive function with which the solution structure, i.e., vehicle-target allocation, is retrieved.

**Algorithm 1**: DP for finding agent-target allocations and corresponding velocities.

**Input**: `The agent set, locations and velocities, the target set and locations, and the distance function;`

`Compute the length of arc between every pair agent-target` $(d_{ij})$ `considering the best arc that avoids obstacles;`

`Label all states as not yet computed;`
$f(n, V, S) = \infty$ ;
`for all` $n = 1, 2, \ldots, N$, `all` $V$ `with` $n$ `components,` $S \in J$;

`Initialize states at stage one as`

$$f(1, V, \{j\}) = \{d_{1j}/v_1\}, \quad \forall V \in \Upsilon, j \in J.$$

`Call` $Compute(N, V, J)$ `for all sequences` $V$ `with` $N$ `components;`
**Output**: `Solution performance;`

`Call` $Allocation(N, V*, J)$;
**Output**: `Agent targets and velocities;`

**Algorithm 2** is a recursive algorithm that computes the optimal solution cost, i.e., it implements equation (8.2). This function receives three arguments: the vehicles to be allocated, their respective velocity values, and the set of target locations available to them, all represented by integer numbers. It starts by checking whether the specific state $(i, V, S)$ has already been computed or not. If so, the program returns to the point where the function was called, otherwise the state is computed. To compute state $(i, V, S)$, all possible target locations $j \in S$ that might lead to a better subproblem solution are identified. The function is then called with arguments $(i-1, V', S')$, where $V' = V\ DIV\ nvel$ (subsequence of $v$ containing the first $i-1$ elements and $S' = S \setminus \{j\}$, for every $j$ such that allocating vehicle $i$ to target $j$ does not lead to any collision with previously defined allocations. This condition is verified by **Algorithm 3**.

**Algorithm 2**: Recursive function: compute optimal performance.

```
Recursive Compute(i, V, S);
if f(i, V, S) ≠ ∞ then
    return f(i, V, S) to caller;
end
Set min = ∞;
for each j ∈ S' do
    S' = S \ {j}; V' = V DIV nvel; v_i = V MOD nvel;
    Call Collision(i, v_i, j, i − 1, V', S')
    if Col(i, j, i − 1, S') = 0 then
        Call Compute(i − 1, V', S');
        t_ij = d_ij/v_i;
        aux = max(f(i − 1, V', S'), t_ij);
        if aux ≤ min then
            min = aux; best_j = j;
        end
    end
end
Store information: target  Be_j(i, V, S) = best_j; value  f(i, V, S) = min;
Return: f(i, V, S);
```

**Algorithm 3** is a recursive algorithm that checks the collision of a specific vehicle-target allocation traveling at a specific velocity with the set of allocations and velocities previously established, i.e., it implements equation (8.5). This function receives six arguments: the newly defined vehicle-target allocation $i \longrightarrow j$ and its traveling velocity $v_i$ and the previously defined allocations and respective velocities to check with, that is vehicles $1, 2, \ldots, k$, their velocities and their target locations $S$. It starts by checking the collision condition, given by equation (8.1), for the pair $i \longrightarrow j$ traveling at velocity $v_i$ and $k \longrightarrow Be_j$ traveling at velocity $v_k$, where $Be_j$ is the optimal target for vehicle $k$ when vehicles $1, 2, \ldots, k$ are allocated to targets in $S$. If there is collision it returns 1; otherwise it calls itself with arguments $(i, v_i, j, k-1, V', S \setminus \{Be_j\})$.

**Algorithm 3**: Recursive function: find if the trajectory of the allocation $i \rightarrow j$ at velocity $v_i$ collides with any of the existing allocations to the targets in $S$ at the specified velocities in $V$.

```
Recursive Collision(i, v_i, j, k, V, S);

if Col(i, v_i, j, k, V, S) ≠ ∞ then
    return Col(i, v_i, j, k, V, S) to caller;
end
B_j = Be_j(k, V, S);
if collision condition is not satisfied then
    Col(i, v_i, j, k, V, S) = 1;
    return Col(i, v_i, j, k, V, S) to caller;
end
S' = S \ {B_j};
V' = V DIV nvel;
v_k = V MOD nvel;
Call Collision(i, v_i, j, k − 1, V', S');

Store information:  Col(i, v_i, j, k, V, S) = 0;

Return:  Col(i, v_i, j, k, V, S);
```

**Algorithm 4** is also a recursive algorithm and it backtracks through the information stored while solving subproblems, in order to retrieve the solution structure, i.e., the actual vehicle-target allocation and vehicle velocity. This algorithm works backward from the final state $(N, V^*, J)$, corresponding to the optimal solution obtained, and finds the partition by looking at the vehicle traveling velocity $v_N = V^*\ MOD\ nvel$ and at the target stored for this state $Be_j\,(N, V^*, J)$, with which it can build the structure of the solution found. **Algorithm 4** receives three arguments: the vehicles, their traveling velocity, and the set of target locations. It starts by checking whether the vehicle current locations set is empty. If so, the program returns to the point where the function was called. Otherwise the backtrack information of the state is retrieved and the other needed states evaluated.

**Algorithm 4**: Recursive function: retrieve agent-target allocation and agents velocity.

```
Recursive Allocation(i, V, S);

if S ≠ ∅ then
    v_i = VMODmvel;
    j = targetBe_j(i, V, S);
    Vloc(i) = v_i;
    Alloc(i) = j;
    V' = VDIVnvel;
    S' = S \ {j};
    CALL Allocation(i − 1, V', S');
end

Return: Alloc;
```

## 8.5 Examples

An example is given to show how vehicle-target allocations are influenced by imposing that no collisions are allowed both with a single fixed velocity value for all vehicles and with the choice of vehicle velocities from 3 different possible values. In this example, we have decided to use $d_{ij}$ as the Euclidian distance although any other distance measure may have been used.

The separation constraints impose, at any point in time, the distance between any two vehicle trajectories to be at least 15 points; otherwise it is considered that those two vehicles collide.

Consider four vehicles, $A$, $B$, $C$, and $D$ with random initial positions as given in Table 8.1 and four target positions 1, 2, 3, and 4 in a diamond formation as given in Table 8.2.

In Figure 8.12 and Figure 8.13 we give the graphical representation of the optimal vehicle-target allocation found, when a single velocity value is considered and collisions are allowed and no collisions are allowed, respectively.

| | **Location** | |
|---|---|---|
| | $\mathbf{x}_i$ | $\mathbf{y}_i$ |
| **Agent A** | 35 | 185 |
| **Agent B** | 183 | 64 |
| **Agent C** | 348 | 349 |
| **Agent D** | 30 | 200 |

Table 8.1: Vehicles random initial location

| | **Location** | |
|---|---|---|
| | $\mathbf{x}_i$ | $\mathbf{y}_i$ |
| **Target 1** | 95 | 258 |
| **Target 2** | 294 | 258 |
| **Target 3** | 195 | 169 |
| **Target 4** | 195 | 347 |

Table 8.2: Target locations in diammond formation.

As it can be seen Figure 8.12, i.e., when collisions are allowed, the trajectory of vehicles A and D do not remain apart, by 15 points, at all times. Therefore, when no collisions are enforced the vehicle-target allocation changes with an increase in the time that it takes for all vehicles to assume their new positions.

In Figure 8.14 and Figure 8.15 we give the graphical representation of the optimal vehicle-target allocation found, when there are 3 possible velocity values to choose from and collisions are allowed and no collisions are allowed, respectively.

As it can be seen Figure 8.14, i.e. when collisions are allowed, the trajectory of vehicles A and D do not remain apart, by 15 points, at all times, since the vehicles move at the same velocity. Therefore, when no collisions are enforced although the vehicle-target allocation remains the same, vehicle A has it velocity decreased and therefore its trajectory no longer collides with the trajectory of vehicle D, see Figure 8.15

Furthermore, since vehicles A trajectory is smaller this can be done with no increase in the time that it takes for all vehicles to assume their new positions.

## 8.6 Notes at the end of Chapter

In this Chapter we have developed an optimization algorithm to decide how to reorganize a formation of vehicles into another formation of different shape with collision avoidance and vehicle traveling velocity choice and with obstacles, which is a relevant problem in cooperative control applications. The method proposed here should be seen as a component of a framework for multivehicle coordination/cooperation, which must necessarily include other components such as a trajectory control component.

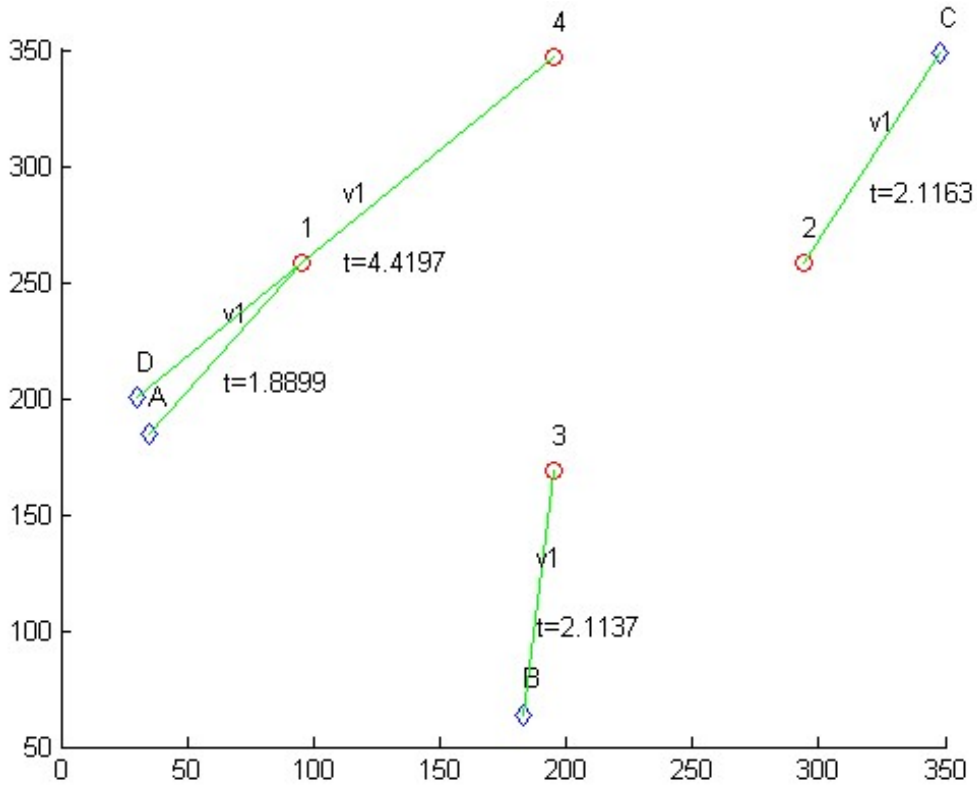


Figure 8.12: A single velocity value is considered and collisions are allowed.

The algorithm proposed is based on a dynamic programming approach that is very efficient for small dimensional problems. As explained before, the original problem is solved by combining, in an efficient way, the solution to some subproblems. The method efficiency improves with the number of times the subproblems are reused, which obviously increases with the number of feasible solutions.

Moreover, the proposed methodology is very flexible, in the sense that it easily allows for the inclusion of additional problem features, e.g., imposing geometric constraints on each vehicle or on the formation as a whole, using nonlinear trajectories, among others.

Initial results for a simplified version of this problem have been publish in [FF08b] and collision avoidance in [FF10]. Allowing variable agent velocities has been done in [FFC11]. The obstacle avoidance improvement was presented at the Optimization 2011 Conference and is submited for publication [CFF12].

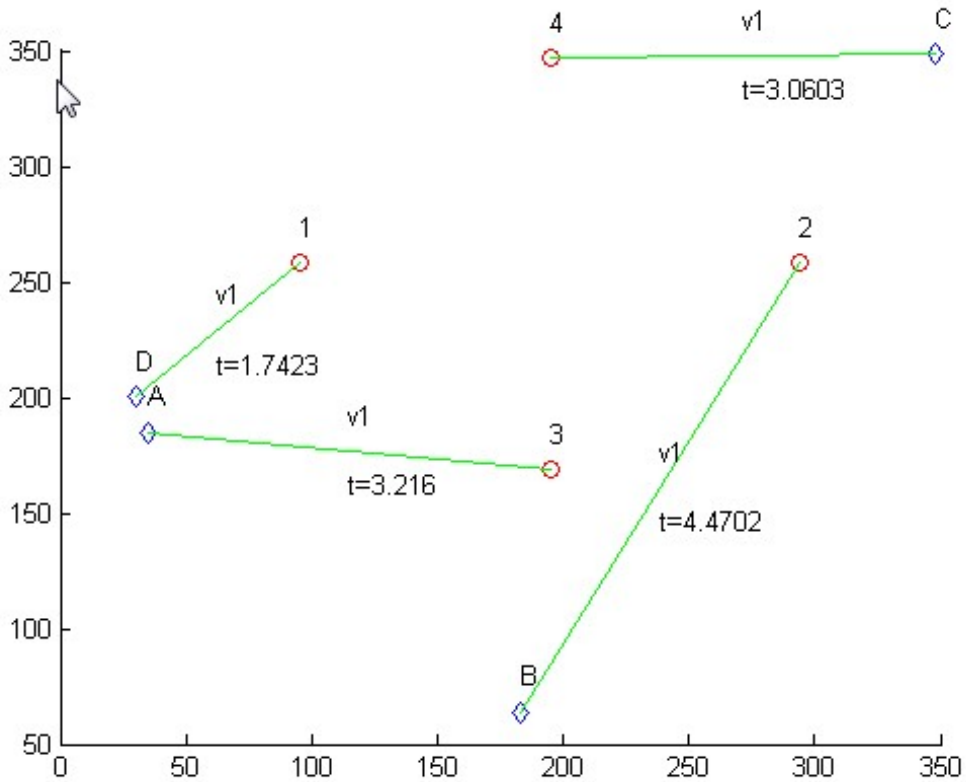


Figure 8.13: A single velocity value is considered and no collisions are allowed.

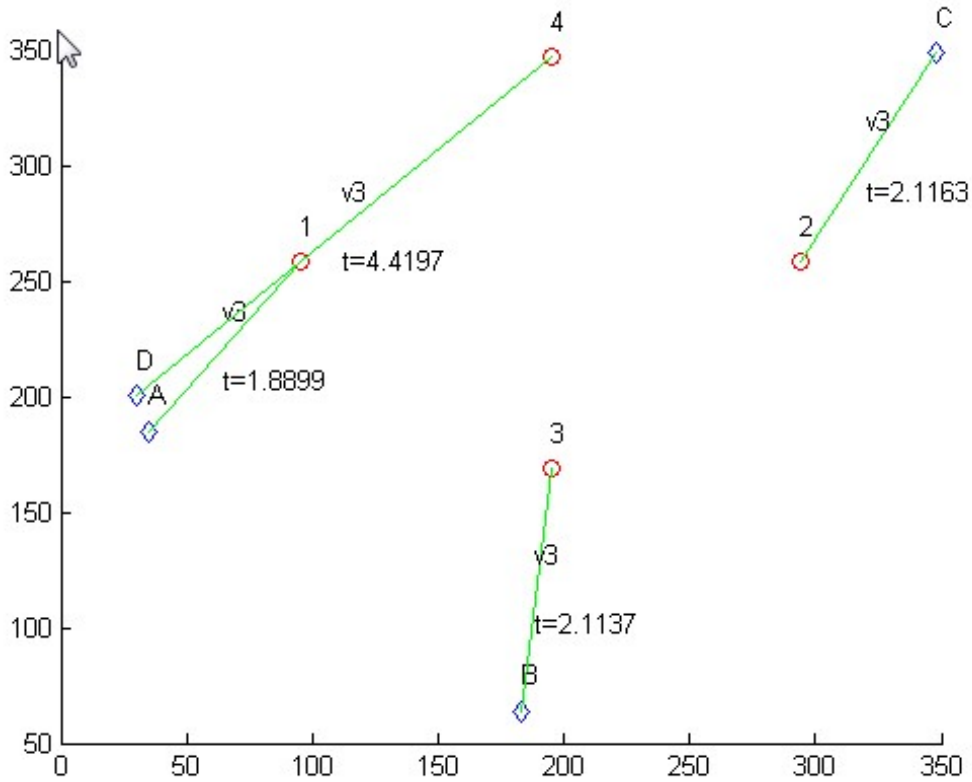


Figure 8.14: Three possible velocity values to choose and collisions are allowed.

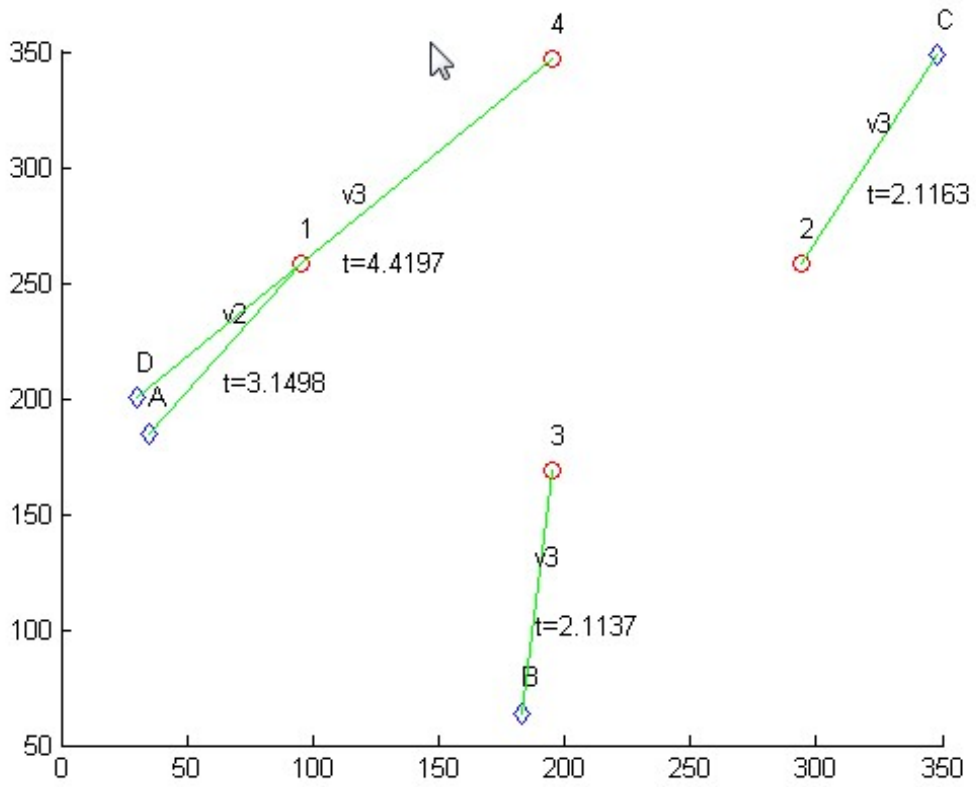


Figure 8.15: Three possible velocity values to choose and no collisions are allowed.

# Chapter 9

# Conclusions and future work

## 9.1 Conclusions

The research described in this thesis has concerned the control of nonholonomic vehicles, in particular, wheeled mobile robots, using optimization-based techniques, especially, Optimal Control and Model Predictive Control. This thesis also deals with formation of mobile robots. The research reported in this dissertation responds to the growing need for collaborative multi-robot systems. In this field, we studyed the control to maintain a given formation and also aspects related to the reorganization of the formation. Once again, optimization is the backbone of this thesis, in both the control and reorganization of the formations.

First we review the key ingredients of this work: nonholonomic systems (in particular wheeled mobile robots), Optimal Control and the Model Predictive Control. All this served as a bridge to the following chapters.

The results on stability proved in [FMG07] use an optimal control problem where the controls are functions selected from a very general set (the set of measurable functions taking values on a set $U$, subset of $\mathbb{R}^m$). This is adequate to prove theoretical stability results and it even permits to use the results on existence of a minimizing solution to optimal control problems (see e.g. [Fon01]). To make computations viable the control functions must be parameterized in same way and so make the problem computationally tractable. That is, for implementation, using any optimization algorithm, the control functions need to be described by a finite number of parameters (the so called *finite parameterizations of the control functions*). The control can be parameterized, for example, as: piecewise constant controls, see e.g. [MS04]; polynomials or splines described by a finite number of coefficients; bang-bang controls, see e.g. [FM03a]. In this theseis, we reported results on the implementation of a stable MPC strategy for a wheeled robot, where finite parameterizations for discontinuous control functions are used, resulting in efficient computation of the control strategies.

Always using as our model the differential-drive mobile robot car (see Figure 2.4), we reported results on the implementation of a stabilizing MPC strategy. Conditions under which steering to a set is guaranteed were established. A set of design parameters satisfying all these

conditions for the control of a unicycle mobile robot were derived. We also discussed the use of MPC to address the problem of path-following of nonholonomic systems. That MPC can solved this problem in a effective and relatively easy way, and has several advantages relative to alternative approaches. The path-following problem was converted into a trajectory-tracking problem and the speed profile at which the path is followed inside the optimization problems solved in the MPC algorithm was determined. Our MPC framework solved a sequence of optimization problems that find an initial point, a speed profile, and a feedback control to track the trajectory of a virtual reference vehicle. That is, the MPC framework finds a feedback control to follow the path given.

In the problem of controlling a vehicle formation, the constraints on the trajectory are an important problem characteristic (to avoid collisions between the vehicles in the formation; to avoid obstacles in the path; and, for other reasons of safety or reliability of operation). Therefore, constraints should be appropriately dealt with. The results included in this work showed that the Model Predictive Control was an adequate tool for control of vehicles in a formation: it deals explicitly and effectively with the constraints that are an important problem feature; recent results on parameterized approaches to the optimal control problems allow addressing fast systems as well as allowing efficient implementations. In this Thesis a control scheme for a set of vehicles moving in a formation was proposed. The control methodology selected was a two-layer control scheme where each layer is based on MPC. The first layer, the trajectory controller, was a nonlinear controller since most vehicles are nonholonomic systems and require a nonlinear, even discontinuous, feedback to stabilize them. The trajectory controller, a model predictive controller, computes control law and only a small set of parameters needs to be transmitted to each vehicle at each iteration. The second layer, the formation controller, aims to compensate for small changes around a nominal trajectory maintaining the relative positions between vehicles. The formation control can be, in most cases, adequately carried out by a linear model predictive controller accommodating input and state constraints. This has the advantage that the control laws for each vehicle are simple piecewise affine feedback laws that can be pre-computed off-line and implemented in a distributed way in each vehicle.

Aspects related to the reorganization of the formation were studied in this thesis. We studied the problem of switching the geometry of a formation of undistinguishable vehicles by minimizing some performance criterion. We have developed an optimization algorithm to decide how to reorganize a formation of vehicles into another formation of different shape with collision avoidance and vehicle traveling velocity choice. Moreover, each vehicle could also modify its path by changing its curvature. The formation switching performance was given by the time required for all vehicles to reach their new position, which was given by the maximum traveling time amongst individual vehicle traveling times. Since we wanted to minimize the time required for all vehicles to reach their new position, we solved a minmax problem. However, our methodology can be used with any separable performance function. The algorithm proposed is based on a dynamic programming approach.

## 9.2 Future work

The work presented in this dissertation opens many opportunities to expand and further develop the field of the optimization and control of nonholonomic vehicles and vehicles formation. The research described here naturally leads on to several open questions and suggests some future developments.

Although we focused on Chapter 6 in a MPC framework that uses a control Lyapunov function (CLF) [Fon01], known as CLF MPC scheme we can do a MPC framework that uses a controllability assumption in terms of the stage cost instead [RA11], known as Unconstrained MPC scheme and compare the two MPC schemes.

We can use the implementations made on Chapter 5 and use dynamic optimizers available (e.g., RIOTS [Sch98], DIDO [Ros07], GPOPS [RDG+], PROPT [RE10] or ACADO [HF11]) and compare all of them.

In this Chapter we treat the control problem for nonholonomic wheeled mobile robot moving on the plane, making the classification of the possible motion tasks as follows [LOS98]: point-to-point motion (a desired goal configuration must be reached starting from a given initial configuration); path following (the robot must reach and follow a geometric path in the cartesian space starting from a given initial configuration (on or off the path)); trajectory tracking (a reference point on the robot must reach and follow a trajectory in the Cartesian space). But our framework limited our analysis to the case of a robot workspace free of obstacles. To be useful in the outside world, the robots must be able to roam freely in large dynamic environments, where there are obstacles and paths unstructured and uncertain, so, we can expand our work considering that there are obstacles on the horizon of the robot, in the first instance these should be static and a second approach (harder) they would be dynamic. In the presence of obstacles, it is necessary to plan motions that enable the robot to execute the assigned task without colliding with them. This problem is referred to as motion planning.

In this thesis we also reported some results on the implementation of stabilizing MPC strategies to a wheeled robot. So, we concentrated our efforts in studying stability, know we can take a jump for studying the also important properties of robustness and performance of the MPC frameworks.

When, in Chapter 7, we propose a two-layer scheme to control a set of vehicles moving in a formation in an environment free of obstacles. Once again we can think of this problem considering an environment more like the real world where there are obstacles (either static or dynamic).

The last part of of this thesis was devoted to developing an optimization algorithm to decide how to reorganize a formation of vehicles into another formation of different shape with collision avoidance and vehicle traveling velocity choice and with obstacles, which is a relevant problem in cooperative control applications. This problem was not treated as a control problem. Therefore the method proposed here should be seen as a component of a framework for multiagent

coordination/cooperation, which must necessarily include other components such as a trajectory control component. So, we can guide our research in this direction. Furthermore, we assumed that each vehicle is holonomic, so we can extend our work to nonholonomic vehicles.

# Bibliography


[AF66] M. Athans and P. L. Falb. *Optimal Control.* McGraw Hill, New York, 1966. Reprinted in 2006 by Dover Books on engineering.

[AHK05] A. P. Aguiar, J. P. Hespanha, and P. V. Kokotovic. Path-following for nonminimum phase systems removes performance limitations. *IEEE Transactions on automatic control*, 50(2):234–239, 2005.

[AHU83] A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *Data Structures and Algorithms.* Addison-Wesley, 1983.

[Ala06a] M. Alamir, editor. *Proceedings of the IFAC Workshop on Nonlinear Model Predictive Control for Fast Systems*, Grenoble, France, October 2006. IFAC, CNRS - Centre National de Recherche Scientifique.

[Ala06b] M Alamir. *Stabilization of Nonlinear Systems Using Receding-Horizon Control Schemes: A Parameterized Approach for Fast System.* Lecture Notes in Control and Information Sciences. Springer, London, 2006.

[Ast95] A. Astolfi. Exponential stabilization of a car-like vehicle. In *Conference on Robotics and Automation*, Nagoya, Japan, 1995.

[Ast96] A. Astolfi. Discontinuous control of nonholonomic systems. *Systems and Control Letters*, 27:37–45, 1996.

[Bel57] R. Bellman. *Dynamic Programming.* Princeton University Press, Princeton, 1957.

[Ben91] J.G. Bender. An overview of systems studies of automated highway systems. *IEEE Transactions on Vehicular Technology*, 40(1 Part 2):82–99, 1991.

[Bet01] J. T. Betts. *Practical Methods for Optimal Control Using Nonlinear Programming.* SIAM, 2001.

[BGW90] R. R. Bitmead, M. Gevers, and V. Wertz. *Adaptive Optimal Control: The Thinking Man's GPC.* Prentice Hall, New York, 1990.

[BMDP02] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos. The explicit linear quadratic regulator for constrained systems. *Automatica*, 38:3–20, 2002.

[BPA93] L. E. Buzogany, M. Pachter, and J. J. Azzo. Automated control of aircraft in formation flight. In *AIAA Guidance, Navigation, and Control Conference and Exhibit, Monterey, California*, pages 1349–1370, 1993.

[Bro83] R. W. Brockett. Asymptotic stability and feedback stabilization. In R. W. Brockett, R. S. Millman, and H. S. Sussmann, editors, *Differential Geometric Control Theory*, pages 181–191. Birkhouser, Boston, 1983.

[Bry96] A. E. Bryson. Optimal control — 1950 to 1985. *IEEE Control Systems*, pages 26–33, June 1996.

[CA98a] H. Chen and F. Allgower. Nonlinear model predictive control schemes with guaranteed stability. In R. Berber and C. Kravaris, editors, *Nonlinear Model Based Process Control*. Kluwer, 1998.

[CA98b] H. Chen and F. Allgower. A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability. *Automatica*, 34(10):1205–1217, 1998.

[CB04] E. F. Camacho and C. Bordons. *Model Predictive Control*. Springer, London, 2nd edition, 2004.

[CF07] A. C. D. Caldeira and F. A. C. C. Fontes. Model predictive control of underactuated mechanical systems. In *Robomat 07*, pages 189–195, 2007.

[CF10] A. C. D. Caldeira and F. A. C. C. Fontes. Model predictive control for path-following of nonholonomic systems. In *Controlo 2010 - 9th Portuguese Conference on Automatic Control*, pages 720–725, 2010.

[CFF12] A. C. D. Caldeira, D. B. M. M. Fontes, and F. A. C. C. Fontes. Optimal switching of agent formations with collision avoidance and allowing variable path curvatures. presented at the Conference Optimization 2011, Costa da Caparica, Portugal; also submitted to an international journal, 2012.

[Cha07] B. Chachuat. *Nonlinear and Dynamic Optimization: From Theory to Practice*. Automatic Control Laboratory, EPFL, Switzerland, 2007.

[Cla83] F. H. Clarke. *Optimization and Nonsmooth Analysis*. Wiley, New York, 1983. reprinted as vol. 5 of Classics in Applied Mathematics, SIAM, Philadelphia, 1990.

[Cla89] F. H. Clarke. *Methods of Dynamic and Nonsmooth Optimization*. SIAM, 1989.

[Cla01] F. H. Clarke. Nonsmooth analysis in control theory: a survey. *European Journal of Control; Special issue: Fundamental Issues in Control*, 7:145–159, 2001.

[CLSS97] F. H. Clarke, Y. S. Ledyaev, E. D. Sontag, and A. I. Subbotin. Asymptotic controllability implies feedback stabilization. *IEEE Transactions on Automatic Control*, 42(10):1394–1407, 1997.

[CLSW98] F. H. Clarke, Y. S. Ledyaev, R. J. Stern, and P. R. Wolenski. *Nonsmooth Analysis and Control Theory.* Springer Verlag, New York, 1998.

[CR94] J.M. Coron and L. Rosier. A relation between time-varying and discontinuous feedback stabilization. *Journal of Math. Systems, Estimation and Control*, 4:67–84, 1994.

[DBS05] M. Diehl, H. G. Bock, and J. P. Schlöder. A real-time iteration scheme for nonlinear optimization in optimal feedback control. *SIAM Journal on Control and Optimization*, 43(5):1714–1736, 2005.

[DL71] P. Dorato and A. Levis. Optimal linear regulators: The discrete-time case. *IEEE Transactions on automatic control*, AC-16(6):613–620, 1971.

[DM06] W. B. Dunbar and R. M. Murray. Distributed receding horizon control for multi-vehicle formation stabilization. *Automatica*, 42:549 – 558, 2006.

[DOK01] J. P. Desai, P. Ostrowski, and V. Kumar. Modeling and control of formations of nonholonomic mobile robots. *IEEE Transactions on Robotics and Automation*, 17(6):905–908, 2001.

[FA03] R. Findeisen and F. Allgöwer. The quasi-infinite horizon approach to nonlinear model predictive control. In A. Zinober and D. Owens, editors, *Nonlinear and Adaptive Control*, volume 281 of *Lecture Notes in Control and Information Sciences*, pages 89–108. Springer Verlag, Berlin, 2003.

[FF08a] T. Faulwasser and R. Findeisen. Nonlinear model predictive path-following control. In *Lecture Notes in Control and Information Sciences*, Berlin, 2008.

[FF08b] D. B. M. M. Fontes and F. A. C. C. Fontes. Optimal reorganization of agent formations. *WSEAS Transactions on Systems and Control*, 3(9):789–798, 2008.

[FF10] F. A. C. C. Fontes and D. B. M. M. Fontes. Minimal switching time of agent formations with collision avoidance. In P. Pardalos M.J. Hirsch and R. Murphey, editors, *Dynamics of Information Systems Theory and Applications*, volume 40, pages 305–321. Springer Optimization and Its Applications Series, Berlin, 2010.

[FFC09] F. A. C. C. Fontes, D. B. M. M. Fontes, and A. C. D. Caldeira. Model predictive control of vehicle formations. In C.W. Commander P. Pardalos M.J. Hirsch and R. Murphey, editors, *Optimization and Cooperative Control Strategies, Lecture Notes in Control and Information Sciences*, volume 381, pages 371–384. Springer Verlag, 2009.

[FFC11] D. B. M. M. Fontes, F. A. C. C. Fontes, and A. C. D. Caldeira. Optimal formation switching with collision avoidance and allowing variable agent velocities. In C.W. Commander P. Pardalos M.J. Hirsch and R. Murphey, editors, *Optimization and Cooperative Control Strategies, Lecture Notes in Control and Information Sciences*, volume '. Springer Verlag, Berlin, 2011.

[FIAF03] R. Findeisen, L. Imsland, F. Allgöwer, and B. Foss. State and output feedback nonlinear model predictive control: an overview. *European Journal of Control*, 9:190–206, 2003.

[Fil88] A. F. Filippov. *Differential Equations with Discontinuous Righthand Sides*. Kluwer Academic, Dordrecht, 1988.

[FM03a] F. A. C. C. Fontes and L. Magni. Min-max model predictive control of nonlinear systems using discontinuous feedbacks. *IEEE Transactions on Automatic Control*, 48:1750–1755, 2003.

[FM03b] F. A. C. C. Fontes and L. Magni. A sufficient condition for robust stability of a min-max MPC framework. Report C1, Officina Mathematica, Universidade do Minho, 4800-058 Guimaraes, Portugal, 2003.

[FMG07] F. A. C. C. Fontes, L. Magni, and E. Gyurkovics. Sampled-data model predictive control for nonlinear time-varying systems: Stability and robustness. In F. Allgower, R. Findeisen, and L. Biegler, editors, *Assessment and Future Directions of Nonlinear Model Predictive Control*, volume 358 of *Lecture Notes in Control and Information Systems*, pages 115–129. Springer Verlag, 2007.

[Fon99] F. A. C. C. Fontes. *Optimisation-Based Control of Constrained Nonlinear Systems*. PhD thesis, Imperial College of Science Technology and Medicine, University of London, London SW7 2BY, U.K., 1999.

[Fon00] F. A. C. C. Fontes. Overview of nonlinear model predictive control schemes leading to stability. In *Proceedings of CONTROLO'2000 – 4th Portuguese Conference on Automatic Control*, GuimarÃčes, Portugal, October 2000.

[Fon01] F.A.C.C. Fontes. A general framework to design stabilizing nonlinear model predictive controllers. *Systems & Control Letters*, 42:127–143, 2001.

[Fon02a] F. A. C. C. Fontes. Discontinuous feedbacks, discontinuous optimal controls, and continuous-time model predictive control. *IJRNC - International Journal of Robust and Nonlinear Control*, 2002. Submitted.

[Fon02b] F. A. C. C. Fontes. Stabilizing nonholonomic systems by model predictive control. In *Controlo 2002 - 5th Portuguese Conference on Automatic Control*, 2002.

[Fon03] F. A. C. C. Fontes. Discontinuous feedbacks, discontinuous optimal controls, and continuous-time model predictive control. *International Journal of Robust and Nonlinear Control*, 13(3–4):191–209, 2003.

[GH05] D. B. Gu and H. S. Hu. A stabilizing receding horizon regulator for nonholonomic mobile robots. *IEEE Transactions on Robotics*, 21(5):1022–1028, 2005.

[GH06] D. B. Gu and H. S. Hu. Receding horizon tracking control of wheeled mobile robots. *IEEE Transactions on Control Systems Technology*, 14(4):743–749, 2006.

[GPSK07] R. Ghabcheloo, A. Pascoal, C. Silvestre, and I. Kaminer. Non-linear co-ordinated path following control of multiple wheeled robots with bidirectional communication constraints. *International Journal of Adaptive Control and Signal Processing*, 21:133–157, 2007.

[GPSW10] L. Grune, J. Pannek, M. Seehafer, and K. Worthmann. Analysis of unconstrained nonlinear mpc schemes with varying control horizons. *SIAM Journal of Control and Optimization*, 48(8):4938–4962, 2010.

[Gru09] L. Grune. Analysis and design of unconstrained nonlinear MPC schemes for finite and infinite dimensional systems. *SIAM Journal of Control and Optimization*, 48(2):1206–1228, 2009.

[GSD04] G. C. Goodwin, M. M. Seron, and J. A. De Dona. *Constrained Control and Estimation: An Optimisation Approach.* Springer, London, 2004.

[GvLPW10] L. Grune, M. von Lossow, J. Pannek, and K. Worthmann. Mpc: implications of a growth condition on exponentially controllable systems. In *8th IFAC Symposium on Nonlinear Control Systems (NOLCOS2010)*, pages 385–390, 2010.

[HF11] Boris Houska and Hans Joachim Ferreau. ACADO. http://www.acadotoolkit.org/, 2011.

[HS01] J. Hu and S. Sastry. Optimal collision avoidance and formation switching on riemannian manifolds. *IEEE Conference on Decision and Control*, 2:1071–1076, 2001.

[JSS06] Z. Jin, T. Shima, and C. J. Schumacher. Optimal scheduling for refueling multiple autonomous aerial vehicles. *IEEE Transactions on Robotics*, 22(4):682–693, 2006.

[Kal60] R. E. Kalman. Contributions to the theory of optimal control. *Bol. de Soc. Math. Mexicana*, 1960.

[K.E89] K.E.Atkinson. *An Introduction to Numerical Analysis.* John Wiley, New York, 1989.

[KGBC06] M. Kvasnica, P. Grieder, M. Baotic, and F. J. Christophersen. MPT - Multiparametric Toolbox. Technical report, ETH - Swiss Federal Institute of Technology, Zurich, 2006.

[KM95] I. Kolmanovsky and N.H. McClamroch. Developments in nonholonomic control problems. *IEEE Control Systems*, pages 20–36, December 1995.

[LFFdP11] S. Lopes, F. A. C. C. Fontes, M. M. A. Ferreira, and M.d.R de Pinho. Notes on convex analysis. Faculdade de Engennharia da Universidade do Porto, 2011.

[LLCF09] N. Lau, L. Lopes, G. Corrente, and N. Filipe. Multi-robot team coordination through roles, positionings and coordinated procedures. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems - IROS*, pages 5841–5848, 2009.

[LO95] A. D. Luca and G. Oriolo. Modelling and control of nonholonomic mechanical systems. In A. Kecskemethy J. Angeles, editor, *Kinematics and Dynamics of Multi-Body Systems*, volume 360 of *CISM Courses and Lectures*, pages 277–342. Springer Verlag, 1995.

[LOS98] A. D. Luca, G. Oriolo, and C. Samson. Feedback control of a nonholonomic car-like robot. In J.-P. Laumond, editor, *Robot Motion Planning and Control*, volume 229 of *Lecture Notes in Control and Information Sciences*, pages 171–253. Springer Verlag, London, 1998.

[Mac02] J. M. Maciejowski. *Predictive Control with Constraints*. Prentice Hall, Harlow, England, 2002.

[May01] D. Q. Mayne. Control of constrained dynamic systems. *European Journal of Control; Special issue: Fundamental Issues in Control*, 7:87–99, 2001.

[MLS94] R. M. Murray, Zexiang Li, and S. Shankar Sastry. *Mathematical Introduction to Robotic Manipulation*. CRC Press, http://www.cds.caltech.edu/ murray/mlswiki, 1994.

[MRRS00] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36:789–814, 2000.

[MS82] J. Macky and A. Strauss. *Introduction to Optimal Control Theory*. Springer-Verlag, Berlin, 1982.

[MS97] L. Magni and R. Sepulchre. Stability margins of nonlinear receding horizon control via inverse optimality. *Systems and Control Letters*, 32:241–245, 1997.

[MS04] L. Magni and R. Scattolini. Model predictive control of continuous-time nonlinear systems with piecewise constant control. *IEEE Transactions on Automatic Control*, 49:900–906, 2004.

[Mur07] R. M. Murray. Recent research in cooperative control of multi-vehicle systems. *Journal of Dynamic Systems, Measurement and Control*, 129:571–583, 2007.

[NvdS90] H. Nijmeijer and A. van de Schaft. *Nonlinear Dynamical Control Systems*. Springer-Verlag, Berlin, Germany, 1990.

[OLV02] G. Oriolo, A. D. Luca, and M. Vendittellir. Wmr control via dynamic feedback linerization: design, implementation, and experimental validation. *IEEE Transactions on Control Systems Technology*, 10:835–852, 2002.

[PBGM62] L. S. Pontryagin, V. G. Boltyanskii, R. V. Gamkrelidze, and E. F. Mishchenko. *The Mathematical Theory of Optimal Processes*. Wiley Interscience, New York, 1962.

[Per01] F. Lobo Pereira. Control design for autonomous vehicles: a dynamic optimization perspective. *European Journal of Control*, 14:178–202, 2001.

[PRMM10] G. Pannocchia, J. B. Rawlings, D. Q. Mayne, and W. Marquardt. On computing solutions to the continuous time constrained linear quadratic regulator. *IEEE Transactions on automatic control*, 55(9):2192–2198, 2010.

[QB97] S. J. Qin and T. A. Badgwell. An overview of industrial model predictive control technology. In *Fifth International Conference on Chemical Process Control*, volume 93 of *AIChe Symposium Series*, pages 232–256, 1997.

[QB98] S. J. Qin and T. A. Badgwell. An overview of nonlinear model predictive control applications. In *Nonlinear Model Predictive Control Workshop*, Ascona, Switzerland, 1998.

[QB03] S. Qin and T. Badgwell. An overview of industrial model predictive control technology. *Control Engineering Practice*, 11:733–764, 2003.

[RA11] M. Reble and F. Allgower. Unconstrained nonlinear model predictive control and suboptimality estimates for continuous-time systems. In *Proceedings of the IFAC World Congress 2011*, Milano, Italy, 2011.

[RDG+] Anil V. Rao, Christopher L. Darby, Divya Garg, Geoffrey T. Huntington, David A. Benson, Michael Patterson, Brendan Mahon, and Camila Francolin. GPOPS. http://www.gpops.org/.

[RE10] Per E. Rutquist and Marcus M. Edvall. PROPT. http://tomdyn.com/, 2010.

[RMS07] D. M. Raimondo, L. Magni, and R. Scattolini. Decentralized MPC of nonlinear systems: An input-to-state stability approach. *International Journal Of Robust And Nonlinear Control*, 17:1651–1667, 2007.

[Ros07] I. Michael Ross. *A Beginner's Guide to DIDO (Ver. 7.3) A MATLAB Application Package for Solving Optimal Control Problems.* Elissar LLC, 2007.

[RS06] S. J. Rasmussen and T. Shima. Branch and bound tree search for assigning cooperating UAVs to multiple tasks. In *American Control Conference 2006*, Institute of Electrical and Electronic Engineers. Minneapolis, Minnesota, USANagoya, Japan, 2006.

[RSM+04] S. J. Rasmussen, T. Shima, J. W. Mitchell, A. Sparks, and P. R. Chandler. State-space search for improved autonomous UAVs assignment algorithm. In *IEEE Conference on Decision and Control*, Paradise Island, Bahamas, 2004.

[Rya94] E. P. Ryan. On Brockett's condition for smooth stabilizability and its necessity in a context of nonsmooth feedback. *SIAM Journal of Control and Optimization*, 32:1597–1604, 1994.

[Sar00] R. W. H. Sargent. Optimal control. *Journal of Computational and Applied Mathematics*, 124:361–371, 2000.

[Sch98] Adam L. Schwartz. RIOTS. http://schwartz-home.com/RIOTS/, 1998.

[SCR02] C. J. Schumacher, P. R. Chandler, and S. J. Rasmussen. Task allocation for wide area search munitions via iterative network flow. In *American Institute of Aeronautics and Astronautics, Guidance, Navigation, and Control Conference 2002*, Reston, Virginia, USA, 2002.

[SCR03] C. J. Schumacher, P. R. Chandler, and S. J. Rasmussen. Task allocation for wide area search munitions with variable path length. In *Institute of Electrical and Electronic Engineers, American Control Conference 2003*, New York, USA, 2003.

[SH99] D. Swaroop and JK Hedrick. Constant spacing strategies for platooning in automated highway systems. *Journal of dynamic systems, measurement, and control*, 121(3):462–470, 1999.

[SHL01] T.R. Smith, H. Hanssmann, and N.E. Leonard. Orientation control of multiple underwater vehicles with symmetry-breaking potentials. *In IEEE Conference on Decision and Control*, 5:4598–4603, 2001.

[SL91] J. E. Slotine and W. Li. *Applied Nonlinear Control.* Prentice Hall, New Jersey, 1991.

[SS80] E. D. Sontag and H. J. Sussman. Remarks on continuous feedback. In *IEEE Conference on Decision and Control, Albuquerque*, pages 916–921, 1980.

[SS95] E. D. Sontag and H. J. Sussman. Nonsmooth control-lyapunov functions. In *IEEE Conference on Decision and Control, New Orleans*, pages 2799–2805, 1995.

[SSVO09] Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, and Giuseppe Oriolo. *Robotics - Modelling, Planning and Control*. Springer-Verlag, London, 2009.

[SW97] H. J. Sussman and J. C. Willems. 300 years of optimal control: From the brachystochrone to the maximum principle. *IEEE Control Systems*, pages 32–44, June 1997.

[VEN01] H. Van Essen and H. Nijmeijer. Non-linear model predictive control of constrained mobile robots. In J. L. Martins de Carvalho, F. A. C. C. Fontes, and M. d. R. de Pinho, editors, *Proceedings of the European Control Conference - ECC'01*, pages 1157–1162, Porto, 2001.

[Vin87] R. B. Vinter. Optimal control theory. Lecture Notes, Summer School "Optimization, Control and Related Subjects", Nice, 1987.

[Vin00] R. B. Vinter. *Optimal Control*. Birkhauser, Boston, 2000.

[WCS96] J. D. Wolfe, D. F. Chichka, and J. L. Speyer. Decentralized controllers for unmanned aerial vehicle formation flight. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, pages 3833–3839, 1996.

[YAB01] H. Yamaguchi, T. Arai, and G. Beni. A distributed control scheme for multiple robotic vehicles to make group formations. *Robotics and Autonomous systems*, 36(4):125–147, 2001.

[Yam99] H. Yamaguchi. A cooperative hunting behavior by mobile-robot troops. *The International Journal of Robotics Research*, 18(9):931–940, 1999.

[Yam04] M. Yamagishi. Social rules for reactive formation switching. Technical Report Technical Report UWEETR-2004-0025, Department of Electrical Engineering, University of Washington, Seattle, Washington, USA, 2004.