Application of Pseudospectral Methods for Minimum Lap Time Optimal Control Problems

Nuno Morais

UMinho | 2023

Universidade do Minho
Escola de Ciências

Nuno Daniel Simões Morais

Application of Pseudospectral Methods for Minimum Lap Time Optimal Control Problems

outubro de 2023

Universidade do Minho
Escola de Ciências

Nuno Daniel Simões Morais

Application of Pseudospectral Methods for Minimum Lap Time Optimal Control Problems

Dissertação de Mestrado
em Matemática e Computação
Especialização em Computação

Trabalho efetuado sob a orientação da
Professora Doutora Carolina Paula Baptista Ribeiro
e da
Professora Doutora Sandra Maria Gonçalves de Vilas Boas Jardim

outubro de 2023

Acknowledgements

I would also like to thank Project BREUCA and everyone involved, especially the professors involved in the Project, for introducing me to the topic of trajectory optimization, which allowed me to acquire the knowledge that I needed for the development of the study presented in this thesis.

I would like to thank Professor Carolina Ribeiro for her tremendous support and fellowship.

Lastly, I would like to thank my close family, my best friends and the people involved in my life in the last few months for their support, for helping me to keep my mental sanity and for always encouraging me.

Statement of Integrity

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the University of Minho.

Resumo

As corridas de automóveis são caraterizadas por uma enorme competição entre os construtores de automóveis e os pilotos. De facto, neste desporto, poucos milissegundos são suficientes para ganhar ou perder uma corrida. Para além das caraterísticas do carro e perícia do condutor, as condições da pista são essenciais para a otimização da perfomance do carro em pista. Neste contexto de aplicação, é crucial encontrar as trajetórias ótimas, minimizando o tempo de volta resolvendo problemas de minimização do tempo de volta. Estes são, normalmente, abordados através de problemas de controlo ótimo cuja solução numérica é matematicamente complexa. Esta dissertação apresenta um estudo sobre os fundamentos matemáticos relacionados com os problemas de controlo ótimo e os métodos de resolução dos problemas de tempo mínimo de volta. Para o efeito, foram considerados métodos de colocação direta, permitindo a transformação do problema de controlo ótimo num problema de programação não linear. Os métodos específicos utilizados para a colocação ortogonal direta incluem Legendre-Gauss, Legendre-Gauss-Lobatto e Legendre-Gauss-Radau, sendo que os métodos de Legendre-Gauss e Legendre-Gauss-Radau foram comparados em termos do seu desempenho na resolução do problema de controlo ótimo para otimimização da trajetória. Os resultados obtidos com cada um dos métodos foram analisados, permitindo compreender as diferenças em termos de tempo de execução, do número de iterações necessárias para que o Problema do Controlo Ótimo encontre uma solução ótima (quando as soluções convergem) e do erro global.

Palavras-chave: *Controlo Ótimo, Métodos Diretos, Métodos Pseudo-espectrais, Tempo Mínimo de Volta, Trajetória Ótima.*

Abstract

Car racing are characterized by an enormous competition between car manufacturers and pilots. In fact, for this motor sport, few milliseconds are enough to win or lose a race. Besides the car characteristics, race track conditions are crucial for the optimization of the vehicular performance in a particular competition scenario. In this application field, is crucial to find optimal trajectories for race cars on racetracks, minimizing the lap time through Minimum-Lap-Time planning problems. These problems are commonly addressed by the Optimal Control Problems whose numerical solutions are quite challenging. This thesis presents a study on mathematical background relating to Optimal Control Problems and methods to solve Minimum-lap-time problems. For the purpose, direct collocation methods were considered in order to transform the optimal control problem into a Nonlinear Programming Problem. The specific methods employed for direct collocation included Legendre-Gauss, Legendre-Gauss-Lobatto and Legendre-Gauss-Radau, where Legendre-Gauss and Legendre-Gauss-Radau methods were compared in terms of their performance in solving the time-optimal control problem for vehicle trajectory planning. The results obtained from each method were analyzed, providing insights into the differences in terms of computational time, number of iterations it takes for the Optimal Control Problem to find an optimal solution (when the solutions converge) and the overall error.

Keywords: *Direct Methods, Pseudospectral methods, Minimum Lap Time, Optimal Control, Optimal Trajectory.*

Contents

1 Introduction **1**

1.1 Motivations and Aims 1

1.2 Outline 4

2 Mathematical Background **5**

2.1 Optimal Control Problem 5

2.1.1 Pontryagin Maximum Principle 9

2.2 Indirect Methods 13

2.2.1 Indirect Shooting 13

2.2.2 Indirect Multiple-Shooting 14

2.3 Direct Methods 16

2.3.1 Direct Shooting and Multiple-Shooting Methods 17

2.3.2 Direct Collocation 20

2.4 Hamilton-Jacobi-Bellman 22

3 Optimal Lap Time for a Racing Vehicle **24**

3.1 Track model 24

3.2 Kinematic of the vehicle and force model 25

3.3 Optimal Control Problem Formulation 31

4 Pseudospectral Collocation Methods **36**

4.1 Collocation Points 36

4.2 Pseudospectral Discretizations 40

4.2.1 Collocation at Legendre-Gauss points 41

4.2.2 Collocation at Legendre-Gauss-Lobatto points 42

4.2.3 Collocation at Legendre-Gauss-Radau points 42

5 Lap Time Simulation **44**

5.1 Software implementation 44

5.2 Methods Review 45

5.3 Optimal Control Problem solutions comparison 46

5.3.1 Results overview 49

5.4 Optimal Trajectories 50

5.4.1 Comparison graphics 50

5.5 Comparison of Resulting Optimal Parameters 61

6 Conclusions **63**

List of Figures

1 Optimal Control numerical techniques scheme 2

2 Schematic illustration of problem (2.4)-(2.10). 8

3 Principle of single shooting 17

4 Illustration of multiple shooting 19

5 Illustration of a track segment ant the curvilinear coordinate systems - figure adopted from [7]. 25

6 Illustration of wheel model forces diagram (see in [7]) 26

7 Kamm's circle in [19]. 30

8 LG, LGR and LGL Collocation points in [23] 37

9 Optimal Trajectory of Austin track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) 51

10 Optimal Trajectory of Brandshatch track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) 51

11 Optimal Trajectory of Budapest track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) 51

12 Optimal Trajectory of Catalunya track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) 52

13 Optimal Trajectory of Hockenheim track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) 52

14 Optimal Trajectory of IMS track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) 52

15 Optimal Trajectory of Melbourne track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) 52

16 Optimal Trajectory of Mexico City track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) . . . 53
17 Montreal track with Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) . . . 53
18 Optimal Trajectory of Monza track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) . . . 53
19 Optimal Trajectory of Moscow Raceway track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) . . . 53
20 Optimal Trajectory of Norisring track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) . . . 54
21 Optimal Trajectory of Nuerburgring track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) . . . 54
22 Optimal Trajectory of Oschersleben track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) . . . 54
23 Optimal Trajectory of Sakhir track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) . . . 54
24 Optimal Trajectory of Sao Paulo track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) . . . 55
25 Optimal Trajectory of Sepang track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) . . . 55
26 Optimal Trajectory of Shanghai track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) . . . 55
27 Optimal Trajectory of Silverstone track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) . . . 55
28 Optimal Trajectory of Sochi track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) . . . 56
29 Optimal Trajectory of Spa Francorchamps track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) . . . 56
30 Optimal Trajectory of Spielberg track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) . . . 56
31 Optimal Trajectory of Suzuka track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) . . . 56

32 Optimal Trajectory of Yas Marina track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) . . . 57
33 Optimal Trajectory of Zandvoort track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right) . . . 57
34 Suzuka Full Track with both trajectories . . . 58
35 Suzuka Straight . . . 58
36 Suzuka Corner . . . 58
37 Monza Full Track . . . 59
38 Monza Straight Line . . . 59
39 Monza Corner . . . 59
40 Detail of Monza Corner . . . 59
41 Norisring Full Track with both trajectories . . . 60
42 Norisring Straight . . . 60
43 Norisring Corner . . . 60
44 Oschersleben Full Track . . . 60
45 Oschersleben corner exit zoomed in . . . 60

List of Tables

1 Coordinate and state variables used in the model of the car. 27

2 Values used as the parameters of the vehicle. 47

3 Results obtained using Legendre-Gauss method when running the Optimal Control Problem. 48

4 Results obtained using Legendre-Gauss-Radau method when running the Optimal Control Problem. 49

5 Resulting parameters from Optimal Control Problem using Legendre-Gauss method . . 61

6 Resulting parameters from Optimal Control Problem using Legendre-Gauss-Radau method 62

Chapter 1

Introduction

1.1 Motivations and Aims

The optimal lap time for a race vehicle in racing tracks is a challenging problem. The approach adopted in the formulation of the problem, in particular, has a massive impact on the possibility of achieving a solution for such problem by using numerical techniques to solve them. The fastest lap time for a racing car represents the minimum possible time for a determined vehicle to complete the entire lap, evaluating the longitudinal and lateral controls of the vehicle along a given track [6, 24].

Optimal control is an area of optimization theory that deals with finding a control for a dynamical system over time. It is highly popular and crucial to many applications in Operations Research, Engineering, and Science [5, 25, 14].

Generally, optimal control problems are addressed numerically due to the complexity of the majority of applications. Since Bellman's work in the 1950s (Bellman [3]), numerical techniques have been used to solve optimal control problems.

The vast majority of optimal control problems observed in real-life situations do not have analytical solutions and depend extensively on numerical techniques like the *direct method*, *indirect method*, or *dynamic programming* [5]. Direct methods transform an optimal control problem to a *Nonlinear Programming Problem* (NLP), which allows the low-cost, seamless application of several well-developed and mature NLP optimizers. Direct methods are less accurate due to the fact that they remove adjoint information including the *Hamilton-Jacobi-Bellman equation* [2] and *Pontryagin's Maximal Principle* [20]. Indirect methods transform the optimal control problem into a Two-Point Boundary Value Problem by applying Pontryagin's maximum principle [20]. With the indirect method, feasible optimal trajectories known as extremals are determined by solving a multiple-point boundary-value problem. Then, it is determined if

any of the obtained extremals is a saddle point, local minimum, or local maximum. The specific extremal with the lowest cost is selected from among the locally optimal solutions. In a direct approach, the state and/or control of the optimal control problem are discretized in some way, and the problem is then translated into a nonlinear optimization problem. Following that, the nonlinear programming problem is solved via well-known optimization approaches.

As it can be seen, indirect and direct approaches are results of two opposing perspectives. On one hand, the indirect technique transforms the optimal control problem into a boundary-value problem, therefore solving the problem indirectly (thus the term indirect). The optimal solution is therefore obtained via an indirect technique by solving a set of differential equations that satisfy endpoint and/or interior point requirements. Figure 1 shows the general framework of numerical techniques.

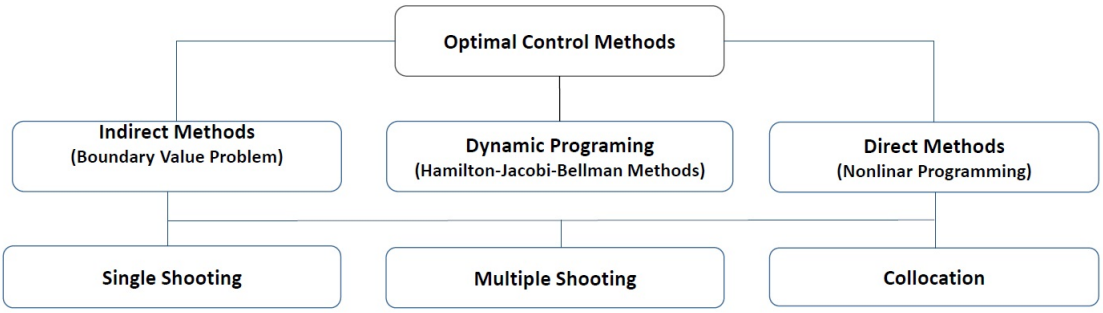

Figure 1: Optimal Control numerical techniques scheme

As reported in [18], direct and indirect methods use one of the following techniques to discretize the dynamical system:

- *A single shooting*: When the trajectory reaches the end time t_f, it is integrated using time-marching techniques starting from t_0. At the conclusion of the integration, boundary restrictions are placed, and the beginning state (and costates) are still unknowns.

- *Multiple shootings*: The time range $[t_0, t_f]$ is divided into N+1 subintervals. The starting values of the state (and adjoints) at each subinterval are unknowns that need to be determined. The trajectory is integrated across each subinterval $[t_i, t_{i+1}]$, $i = 1, \cdots, N$. Additionally, each subinterval's interface needs to impose continuity constraints.

- *Collocation*: The states (and costates) are discretized over a predetermined time-grid so that they are only known at discrete moments. The system-governing equations are converted into discrete defect constraints to establish a connection between the values at the start and end of the

subinterval. The quadrature rule used to estimate the differential equations between each pair of subintervals determines the distinct approaches: both local and global collocation approaches.

Orthogonal collocation methods are a subset of collocation techniques that have been widely used in optimum control. The method used to choose the collocation locations is the main distinction between an orthogonal collocation method and a conventional collocation approach. In particular, the roots of a polynomial that belong to a family of orthogonal polynomials serve as the collocation points in an orthogonal collocation technique, and the polynomial is connected to a quadrature algorithm for approximating the definite integral of a given function. Orthogonal collocation points that come from the roots of either Chebyshev polynomials or Legendre polynomials are common. The *pseudospectral methods* or orthogonal collocation methods, which collocate the differential algebraic equations using nodes derived from a Gaussian quadrature and parameterize the state and control using global polynomials, may be successfully addressed by using optimization tools [11]. *Legendre-Gauss* (LG), *Legendre-Gauss-Radau* (LGR), and *Legendre-Gauss-Lobatto* (LGL) points are the three most often used sets of collocation points.

Several works, such as Massaro and Limebeer [17], Garg et al. [9, 10], show that the orthogonal collocation methods are widely used methods in optimal control of road vehicles, focusing, especially, on minimum lap time simulations.

The objective of any racing car developed by any manufacturer is to do full laps in the least amount of time. To achieve this objective, it is required to determine the trajectory that gives us the best results, taking in account the limitations of the vehicle. In order to do so, we need to get the optimal trajectory and controls for the car that give us in the minimal lap time. To do this, it is used an optimal control problem, which is a mathematical problem, that gives the optimal controls and trajectory for the vehicle, so that the vehicle can do the lap in the least amount of time. And that is the goal of this thesis: to study an optimal control problem already defined, understand the formulation, and then program a new method that gives us the minimal lap time, but with a computational time faster than the method implemented.
In this thesis the track and vehicle models and the optimal control problem formulation used in the paper [7] will be introduced. Also, will be presented the principles of Optimal Control Problem, Pseudospectral collocation method, and an introduction and formulation of three different pseudo-spectral methods.
In the end a comparison between the two pseudospectral methods chosen is made, as well as a comparison between the computational time, optimal trajectory and resulting parameters of the method from the paper [7], This will enable the selection of the pseudospectral method more suitable to be used in finding the time-optimal trajectory for more than 3000 race tracks which will be the input for a machine learning training dataset.

1.2 Outline

A summary of the contents of each chapter is presented in the following paragraphs. After this introductory chapter, **Chapter 2** gives an overview of the Mathematical Theory of optimal control. It presents the basic theory and discusses connections with the Pontryagin Maximum Principle. It also includes a brief introduction to pseudospectral techniques and numerical analysis.

Chapter 3 introduces the minimal-lap-time optimal control problem for a race car, including the track and vehicle models. Then, using the Pontryagin's maximum principle, the adjoint equations are formulated.

Chapter 4 gives an overview on pseudospectral collocation methods for the numerical solution of optimal control problems using the following Gaussian quadrature collocation points: Legendre–Gauss (LG), Legendre–Gauss–Radau (LGR), and Legendre–Gauss–Lobatto (LGL).

Chapter 5 solves the optimal control problem described in **Chapter 4**, using an open-source Python software package for generating global racetrajectories, which can be found on `https://github.com/TUMFTM/global_racetrajectory_optimization`. Then, a comparison between LG and LGR pseudospectral methods, to solve numerically the minimal-lap-time optimal control problem for several race tracks, is presented.

Chapter 6 which presents a summary of the findings, some recommendations for further research and concludes the thesis.

Chapter 2

Mathematical Background

In this chapter, a mathematical background study of optimal control problems and approaches that can be used to solve optimal control problems will be presented. For a more detailed description of mathematical background, the interested reader is referred to [13, 15, 16, 26, 21].

2.1 Optimal Control Problem

The goal of an Optimal Control Problem (OCP) is to find a sequence of control variables that minimize a cost function while adhering to specific constraints. As a result, the problem's constituent aspects are:

- a mathematical model of the system, which is a set of differential equations that describe the vehicle dynamics and aerodynamic constraints;
- a cost function, which reflects a global judgment on the behavior of the system when it is stimulated by control variables with a real value;
- state and control variable constraints.

Dynamics Systems

A dynamic system is a set of differential equations that describe the behavior of complex dynamical systems. In the context of the optimal control theory, the dynamic system may be stated as a constraint as follows:

$$\frac{d\boldsymbol{x}}{dt} = \dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) \quad t \in T \tag{2.1}$$

where $\boldsymbol{x} : T \to \mathbb{R}^n$ and $\boldsymbol{u} : T \to \mathbb{R}^m$ represent the system's state and input control vectors with time horizon T given by $T := [t_0, t_f] \subset \mathbb{R}$. Moreover, the system model dynamics $\boldsymbol{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is composed by a set of Ordinary Differential Equations. The function $\boldsymbol{f}$ is a vector of length n that may be written in matrix form using the conventional state-space representation:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}(t)\boldsymbol{x}(t) + \boldsymbol{B}(t)\boldsymbol{u}(t),$$

where $\boldsymbol{A}(t)$ and $\boldsymbol{B}(t)$ are the state and control matrices.

Cost Function

The objective function $J : \mathcal{X} \times \mathcal{U} \to \mathbb{R}$, in Bolza form, is written as

$$J = \Phi(\boldsymbol{x}(t_f), t_f) + \int_{t_0}^{t_f} L(t, \boldsymbol{x}(t), \boldsymbol{u}(t))dt,$$

where Φ is the terminal cost and L is the *Lagrangian function*, representing the immediate cost at time t as a function of the state $\boldsymbol{x}$ and the control $\boldsymbol{u}$. The set $\mathcal{U}$ contains all measurable control inputs $\boldsymbol{u}$ and $\mathcal{X}$ represents the set of all state trajectories $\boldsymbol{x}$. When $\Phi = 0$, the cost function is in Lagrange form and, if $L = 0$, then it is known as the Mayer's form. The Bolza problem may be simplified to the Mayer problem by adding one additional variable, $z(t)$, which can be expressed in matrix form with the classical state-space representation:

$$\dot{\boldsymbol{y}}(t) = \begin{pmatrix} \dot{\boldsymbol{x}}(t) \\ \dot{z}(t) \end{pmatrix} = \begin{pmatrix} \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) \\ L(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) \end{pmatrix} \tag{2.2}$$

and

$$J = \Phi(t_f, \boldsymbol{x}(t_f)) + (z(t_f) - z(t_0)). \tag{2.3}$$

The cost function indicates the type of problem to be addressed, where the most common formulations are:

1. minimum time control problem - drive a state from $\boldsymbol{x}_0$ to $\boldsymbol{x}_f$ in the shortest possible time

$$J = \int_{t_0}^{t_f} 1\, dt = t_f - t_0 = t^*;$$

2. minimum fuel control problem - move a state from an initial to a final position minimizing the consumption of fuel
$$J = \int_{t_0}^{t_f} |\boldsymbol{u}(t)| dt;$$
3. minimum energy control problem - move from $\boldsymbol{x}_0$ to $\boldsymbol{x}_f$ while minimizing the dissipated energy
$$J = \int_{t_0}^{t_f} \boldsymbol{u}^2(t) dt.$$

Constraints

The following is a general form of a constraint:

$$\phi(t_0, \boldsymbol{x}(t_0), t_f, \boldsymbol{x}(t_f)) = 0,$$

where $\phi(t_0, \boldsymbol{x}(t_0), t_f, \boldsymbol{x}(t_f)) : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^m \to \mathbb{R}^q$ are the boundary conditions and q is their number. The most often used phrase categorizes two groups:

- Initial condition:
$$\boldsymbol{x}(t_0) = \boldsymbol{x}_0;$$
- Boundary Conditions:
$$\Psi(t_f, \boldsymbol{x}(t_f)) = 0.$$

When the end states must approach a certain value, the simplest basic formulation is:

$$\Psi = \boldsymbol{x}(t_f) - \boldsymbol{x}_f = \mathbf{0}.$$

Until now, the only constraints have been placed at the time interval's endpoints, but the true difficulty arises when the solution must fulfill constraints along the route, on the state, or on the control variables. Often, states variables $\boldsymbol{x}$ have only boundaries to obey, i.e.

$$\boldsymbol{x}_{\text{min}} \leq \boldsymbol{x}(t) \leq \boldsymbol{x}_{\text{max}},$$

where $\boldsymbol{x}_{\text{min}}$ and $\boldsymbol{x}_{\text{max}}$ are the maximum and minimum value of $\boldsymbol{x}$; while control variables can also have equality and inequality route constraints:

- Bounds:

$$\boldsymbol{u}_{\text{min}} \leq \boldsymbol{u}(t) \leq \boldsymbol{u}_{\text{max}}$$

where $\boldsymbol{u}_{\text{min}}$ and $\boldsymbol{u}_{\text{max}}$ are the maximum and minimum value of $\boldsymbol{u}$.

- Equality constraints:

$$\boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{u}(t), t) = 0.$$

- Inequality constraints (as an extension of Equality constraints):

$$\boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{u}(t), t) \leq 0.$$

The optimal control problem (see Figure 2) is therefore stated as follows:

$$\text{minimize} \quad J(\boldsymbol{u}) = \Phi(\boldsymbol{x}(t_f), t_f) + \int_{t_0}^{t_f} L(t, \boldsymbol{x}(t), \boldsymbol{u}(t))dt, \tag{2.4}$$

$$\text{subjected to} \quad \dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)), \tag{2.5}$$

$$\phi(t_0, \boldsymbol{x}(t_0), t_f, \boldsymbol{x}(t_f)) = 0 \tag{2.6}$$

$$\Psi(t_f, \boldsymbol{x}(t_f)) = 0, \tag{2.7}$$

$$\boldsymbol{x}_l \leq \boldsymbol{x}(t) \leq \boldsymbol{x}_r, \tag{2.8}$$

$$\boldsymbol{u}_l \leq \boldsymbol{u}(t) \leq \boldsymbol{u}_r, \tag{2.9}$$

$$\boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{u}(t), t) \leq 0. \tag{2.10}$$

The Optimal Control Problem is illustrated in Figure 2

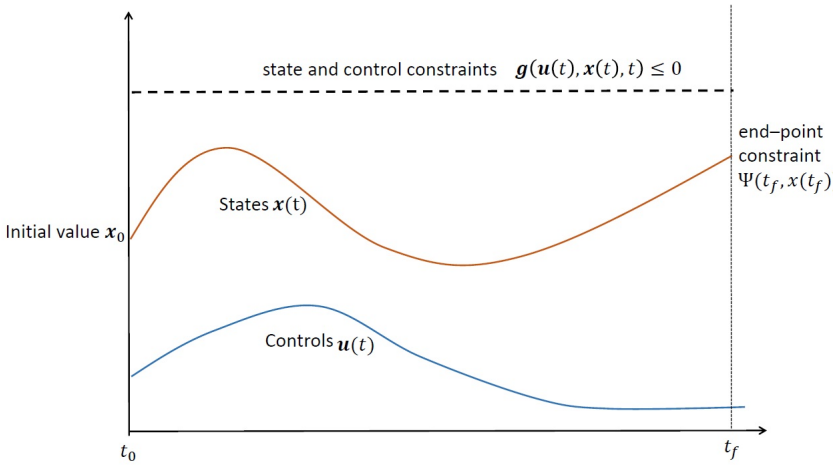

Figure 2: Schematic illustration of problem (2.4)-(2.10).

The inclusion of these different type of constraints can greatly complicate the search for a solution, and different techniques has developed different to cope with it.

2.1.1 Pontryagin Maximum Principle

Pontryagin Maximum Principle is a fundamental concept in the Optimal Control Theory, when one needs to find the best possible control strategy for a dynamical system while optimizing a certain performance criterion, especially in the presence of constraints for the state or input controls. This principle, named after the Russian mathematician Lev Pontryagin [26], states that it is necessary for any optimal control along with the optimal state trajectory to solve a system called a *Hamiltonian system*, which is a two-point boundary value problem, and also a maximum condition of the control Hamiltonian.

This principle seeks to answer the question of how to find the control inputs that maximize (or minimize) a certain objective function, subject to a set of dynamic constraints, which typically take the form of differential equations that describe the evolution of a system over time. The Pontryagin Maximum Principle introduces the concept of a cost functional that quantifies the performance or cost associated with a particular control strategy.

The significance of the maximum principle lies in its ability to transform complex optimization problems into a set of mathematical equations and conditions that can be solved systematically and, in essence, the PMP guarantees that an optimal control strategy must satisfy a set of conditions, including a differential equation known as the Hamiltonian equation and a set of *Transversality Conditions* at the endpoints of the time interval. This principle has been described in detail in [16].

Hamiltonian Function

The Lagrangian approach can be used to augment the cost functional (2.4) with the constraint (2.5), creating the unconstrained problem. To this end, the augmented cost functional $\mathcal{J}$ is introduced:

$$\mathcal{J} = \Phi(\boldsymbol{x}(t_f), t_f) + \int_{t_0}^{t_f} \Big(L(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) + \boldsymbol{\lambda}^T(t)[\boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) - \dot{\boldsymbol{x}}(t)] \Big)\, dt, \tag{2.11}$$

and the Hamiltonian H is defined as

$$H(t, \boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{\lambda}(t)) = L(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) + \langle \boldsymbol{\lambda}^T(t), \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) \rangle,$$

where $\langle \cdot, \cdot \rangle$ denotes the usual dot product and $\boldsymbol{\lambda}$ is the vector of costate variables (or adjoint variables) associated with the state dynamics, representing the sensitivity of the cost function to changes in the state variables.

Pontryagin's Principle of Optimality

This principle follows from the Variational optimal control developed in [16]. The Pontryagin's Principle of Optimality states that the optimal control policy, denoted as $\boldsymbol{u}(t)$, maximizes the Hamiltonian function $H(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}, t)$ at each time. If the set of all admissible controls is denoted by $\mathcal{U}$, this principle states that the resulting optimal control $\boldsymbol{u}^\star$ that minimizes the value of the cost functional $\mathcal{J}$ must satisfy [16]:

$$H(t, \boldsymbol{x}^\star(t), \boldsymbol{u}^\star(t), \boldsymbol{\lambda}^\star(t)) \leq H(t, \boldsymbol{x}^\star(t), \boldsymbol{u}, \boldsymbol{\lambda}^\star(t)), \quad \forall \boldsymbol{u} \in \mathcal{U}, \tag{2.12}$$

for all $t_0 \leq t \leq t_f$. The control that induces the Hamiltonian to respond globally is, thus, the optimal control. Additionally, the idea is frequently stated as

$$\boldsymbol{u}^*(t) = \arg\max_{\boldsymbol{u} \in \mathcal{U}} H(t, \boldsymbol{x}^*(t), \boldsymbol{u}, \boldsymbol{\lambda}^*(t))]. \tag{2.13}$$

In other words, the control input that maximizes the Hamiltonian at each instance of time is the ideal control input. Additionally, if the $\boldsymbol{u}^\star$ lie on the $\mathcal{U}$ border, then (2.12) and (2.13), known as the weak form of the Minimum Principle, can be easily used to identify the best control. On the other hand, if $\boldsymbol{u}^\star$ is included within $\mathcal{U}$, it can be used the strong form of the Minimum Principle, $\nabla_{\boldsymbol{u}} H = \mathbf{0}$, which is also stated as

$$\frac{\partial H}{\partial \boldsymbol{u}}(t, \boldsymbol{x}^*(t), \boldsymbol{u}^*(t), \boldsymbol{\lambda}^*(t)) = \mathbf{0}.$$

Costate Dynamics and Transversality

According to the variational optimal control theory [16], the variational derivative of $\mathcal{J}$ with regard to $\boldsymbol{x}$ and $\boldsymbol{u}$ is given by

$$\delta\mathcal{J} = D\mathcal{J} \cdot \delta\boldsymbol{v} = \Phi_{\boldsymbol{x}}^\top \delta\boldsymbol{x}(t_f) + \int_{t_0}^{t_f} L_{\boldsymbol{x}}^\top \delta\boldsymbol{x} + L_{\boldsymbol{u}}^\top \delta\boldsymbol{u} + \boldsymbol{\lambda}^\top \boldsymbol{f}_{\boldsymbol{x}} \delta\boldsymbol{x} + \boldsymbol{\lambda}^\top \boldsymbol{f}_{\boldsymbol{u}} \delta\boldsymbol{u} - \boldsymbol{\lambda}^\top \delta\dot{\boldsymbol{x}} dt$$

where $\delta\boldsymbol{v} = [\delta\boldsymbol{x}, \delta\boldsymbol{u}]^\top$ is the variation in the state $\delta\boldsymbol{x}$ and control $\delta\boldsymbol{u}$, $D\mathcal{J}$ represents the variational derivative of $\mathcal{J}$ with respect to $\boldsymbol{x}$ and $\boldsymbol{u}$, and the subscripts indicate derivatives terms (i.e., $\Phi_{\boldsymbol{x}} = \frac{\partial \phi}{\partial \boldsymbol{x}}$).

To evaluate the last term on the right, integration by parts is used:

$$-\int_{t_0}^{t_f} \boldsymbol{\lambda}^\top \delta\dot{\boldsymbol{x}}\, dt = -\boldsymbol{\lambda}(t_f)^\top \delta\boldsymbol{x}(t_f) + \boldsymbol{\lambda}(t_0)^\top \delta\boldsymbol{x}(t_0) + \int_{t_0}^{t_f} \dot{\boldsymbol{\lambda}}^\top \delta\boldsymbol{x}\, dt$$

and, therefore, the variation $\delta\mathcal{J}$ becomes:

$$\begin{aligned}\delta\mathcal{J} =& \Phi_{\boldsymbol{x}}(\boldsymbol{x}(t_f))^\top \delta\boldsymbol{x}(t_f) + \int_{t_0}^{t_f} (L_{\boldsymbol{u}}^\top + \boldsymbol{\lambda}^\top \boldsymbol{f}_{\boldsymbol{u}})\delta\boldsymbol{u}\, dt + \\ & \int_{t0}^{t_f} (L_{\boldsymbol{x}}^\top + \boldsymbol{\lambda}^\top \boldsymbol{f}_{\boldsymbol{x}} + \dot{\boldsymbol{\lambda}}^\top)\delta\boldsymbol{x}\, dt - \boldsymbol{\lambda}(t_f)^\top \delta\boldsymbol{x}(t_f) + \boldsymbol{\lambda}(t_0)^\top \delta\boldsymbol{x}(t_0).\end{aligned}$$

Note that $\delta\mathcal{J}$ has three distinct components: state variation ($\delta\boldsymbol{x}$), control variation ($\delta\boldsymbol{u}$), and boundary conditions ($\delta\boldsymbol{x}(t_0), \delta\boldsymbol{x}(t_f)$). If each of these portions of the variation are equal to zero (i.e., $\delta\mathcal{J} = 0$), that would suggest that the solution $(\boldsymbol{u}(t), \boldsymbol{x}(t))$ is at an optimal in terms of a solution for the objective function (to solve the objective function one may set the gradient equal to zero). Let us set each component equal to zero in order to try and solve the system of equations:

$$L_{\boldsymbol{u}} + \boldsymbol{f}_{\boldsymbol{u}}^\top \boldsymbol{\lambda} = 0, \tag{2.14}$$

$$L_{\boldsymbol{x}} + \boldsymbol{f}_{\boldsymbol{x}}^\top \boldsymbol{\lambda} + \dot{\boldsymbol{\lambda}} = 0, \tag{2.15}$$

$$\Phi_{\boldsymbol{x}}(\boldsymbol{x}(t_f)) - \boldsymbol{\lambda}(t_f) = 0. \tag{2.16}$$

The last two previous conditions define the next differential equation:

$$\dot{\boldsymbol{\lambda}} = -L_{\boldsymbol{x}} - \boldsymbol{f}_{\boldsymbol{x}}^\top \boldsymbol{\lambda} = -H_{\boldsymbol{x}} \tag{2.17}$$

$$\text{subject to} \quad \boldsymbol{\lambda}(t_f) = \Phi_{\boldsymbol{x}}(\boldsymbol{x}(t_f)), \tag{2.18}$$

which is solved backwards in time (starting from the boundary condition $\boldsymbol{x}(t_f) = \Phi_{\boldsymbol{x}}$). Note that it is assumed that $\boldsymbol{x}(t_0)$ does not change, therefore the variation $\delta\boldsymbol{x}(t_0) = 0$.

In summary, the first order necessary conditions for the optimal control problem are given by [21],

[17]:

$$\dot{\boldsymbol{x}}^{\star} = \frac{\partial H}{\partial \dot{\boldsymbol{\lambda}}}(t, \boldsymbol{x}^*(t), \boldsymbol{u}^*(t), \boldsymbol{\lambda}^*(t)), \tag{2.19}$$

$$\boldsymbol{\lambda}^{\star} = -\frac{\partial H}{\partial \boldsymbol{x}}(t, \boldsymbol{x}^*(t), \boldsymbol{u}^*(t), \boldsymbol{\lambda}^*(t)), \tag{2.20}$$

$$\boldsymbol{u}^{\star} = \arg\max_{\boldsymbol{u}\in\mathcal{U}}[H(t, \boldsymbol{x}^*(t), \boldsymbol{u}, \boldsymbol{\lambda}^*(t))], \tag{2.21}$$

$$0 = \boldsymbol{x}(t_0), \tag{2.22}$$

$$\boldsymbol{\lambda}_T(t_f) = 0, \tag{2.23}$$

$$\mu_j(t) = 0, \quad \text{when} \quad g_j(t, \boldsymbol{x}^{\star}, \boldsymbol{u}^{\star}) < 0, \quad j = 1, \cdots, n_c, \tag{2.24}$$

$$\mu_j(t) \leq 0, \quad \text{when} \quad g_j(t, \boldsymbol{x}^{\star}, \boldsymbol{u}^{\star}) = 0, \quad j = 1, \cdots, n_c, \tag{2.25}$$

where $\mu(t)$ is the path constraint multiplier, $H(t, \boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{u}, \boldsymbol{\mu}, \boldsymbol{u}) = L + \boldsymbol{\lambda}^T \boldsymbol{f} - \boldsymbol{\mu}^T \boldsymbol{g}$ is the control Hamiltonian. The initial and final costate criteria in equations (2.22)- (2.23) are referred to as *transversality conditions*, whereas the *complementary slackness conditions* are applied to the Lagrange multipliers of the route constraints provided in (2.24)-(2.25). Furthermore, equation (2.21) is a well-known formula for determining the optimal control and is sometimes referred to as *Pontryagin's Minimum Principle*. An *Hamiltonian boundary-value problem* is a mathematical problem that involves the Hamiltonian system as well as the boundary conditions, transversality criteria, and complementary slackness constraints.

An extremal is any solution $(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{\lambda}(t), \boldsymbol{\mu}(t))$ that consists of the state, costate, and any Lagrange multipliers that fulfill the boundary requirements and any interior-point constraints on the state and costate.

The solution of an optimal control problem may be difficult, and numerous strategies have been investigated throughout the years. There are three primary techniques for solving these problems, namely:

1. **Indirect methods** including calculus of Variations, Euler-Lagrange differential equations, and the Maximum Principle;

2. **Direct Methods** based on control parameterization in a finite dimension;

3. **Hamilton-Jacobi-Bellman** Partial Differential Equations (HJB-PDEs) and Dynamic Programming.

2.2 Indirect Methods

The optimal control problem must be used to obtain the first-order optimality necessary indirectly. A boundary-value problem is made up of the transversality requirements and the Hamiltonian system. By applying Equations (2.19)–(2.20), the differential equation in this Boundary-Value Problem is:

$$\dot{\boldsymbol{y}}(t) = \begin{bmatrix} -H_{\boldsymbol{\lambda}} \\ H_{\boldsymbol{x}} \end{bmatrix} \tag{2.26}$$

where the boundary requirements $\boldsymbol{y}(t_0)$ and $\boldsymbol{y}(t_f)$ are partially known from the transversality conditions, and

$$\boldsymbol{y}(t) = \begin{bmatrix} \boldsymbol{x}(t) \\ \boldsymbol{\lambda}(t) \end{bmatrix} \tag{2.27}$$

is the combined state and costate vector.

Extremal trajectories, or the solutions to the Hamiltonian Boundary-Value Problem, are computed numerically using an indirect technique. The original optimal control problem is transformed into the challenge of solving a system of nonlinear equations of the type:

$$\begin{aligned} \boldsymbol{f}(\boldsymbol{y}) &= 0 \\ \boldsymbol{g}_{min} \leq \boldsymbol{g}(\boldsymbol{y}) &\leq \boldsymbol{g}_{max} \end{aligned} \tag{2.28}$$

where $\boldsymbol{f}$ is the algebraic equation and $\boldsymbol{g}$ is a mixed constraints.

The *shooting method,* the *multiple-shooting method*, and *collocation approaches* are the three indirect techniques that are used most frequently. Now let's introduce briefly each of these strategies, following the work in [16].

2.2.1 Indirect Shooting

According to [15, 26], in an indirect shooting method, the initial unknown boundary conditions are initially estimated via a shooting approach at one end of the interval, $\boldsymbol{y}(t_0)$ or $\boldsymbol{y}(t_f)$. Using a time shooting technique, the differential equations in Boundary-Value Problem are integrated to the opposite end (t_f or t_0) to yield the state and the costate trajectory:

$$\hat{\boldsymbol{y}}(t_f) = \boldsymbol{y}(t_0) + \int_{t_0}^{t_f} \dot{\boldsymbol{y}}(t)dt \tag{2.29}$$

or

$$\hat{\boldsymbol{y}}(t_0) = \boldsymbol{y}(t_f) - \int_{t_f}^{t_0} \dot{\boldsymbol{y}}(t)dt \tag{2.30}$$

The given boundary conditions, $\boldsymbol{y}(t_f)$ or $\boldsymbol{y}(t_0)$, are compared with the integrated boundary conditions at the opposite end, $\boldsymbol{y}(t_0)$ or $\boldsymbol{y}(t_f)$. The Boundary-Value Problem is solved if the difference between the given and integrated boundary conditions, $||\hat{\boldsymbol{y}}(t_f)-\boldsymbol{y}(t_f)||$ or $||\hat{\boldsymbol{y}}(t_0)-\boldsymbol{y}(t_0)||$, is less than a given tolerance. If not, the estimate of $\boldsymbol{y}(t_0)$ or $\boldsymbol{y}(t_f)$ unknown boundary condition must be revised until the difference is smaller than the tolerance. Algorithm 2.1 briefly displays a schematic of an indirect shooting technique that is proposed in [15].

Algorithm 2.1 Algorithm for the indirect shooting method

1: **Input**: Initial guess of the unknown boundary conditions at t_0
2: **output**: The optimal boundary conditions and the optimal trajectory
3: **Initialization**: Obtain $\boldsymbol{y}(t_f)$ by eq. (2.29)
4: Main step:
5: **while** $||\hat{\boldsymbol{y}}(t_f)-\boldsymbol{y}(t_f)|| > 0$ tolerance **do**
6: Update the initial guess
7: Obtain $\boldsymbol{y}(t_f)$ by eq. (2.29)
8: **end while**

The algorithm is analogous for t_f. The unknown boundary conditions are initially estimated via a shooting approach at one end of the interval. The Hamiltonian system equation (2.19)-(2.20) is integrated to the opposite end (that is, either forward from t_0 to t_f or backward from t_f to t_0) using this guess along with the known initial circumstances. The terminal conditions obtained by numerical integration are compared to the known terminal conditions given in Eqs. (2.22) and (2.23) when they arrive at t_f.

2.2.2 Indirect Multiple-Shooting

Using a multiple shooting approach, interior points

$$t_0 < t_1 < \cdots < t_K = t_f$$

are selected, dividing the time domain $[t_0, t_f]$ into K subintervals. A single shooting approach can be used inside each subinterval $[t_k, t_{k+1}]$, $k = 0, \cdots, K-1$. It involves integrating the differential equation

as $\boldsymbol{y}(t_k)$ and making an initial prediction about the state and costate at each interior point t_k,

$$\hat{\boldsymbol{y}}(t_{k+1}) = \boldsymbol{y}(t_k) + \int_{t_k}^{t_{k+1}} \dot{\boldsymbol{y}}(t)dt \tag{2.31}$$

The continuity criteria

$$\hat{\boldsymbol{y}}(t_k) = \boldsymbol{y}(t_k)$$

must be kept since the state and costate trajectories are continuous at the interior points t_k, with $k = 1, \cdots, K-1$. Using the resultant system of equations:

$$\boldsymbol{F_m} = \begin{bmatrix} \boldsymbol{y}(t_0) - \boldsymbol{y}_0 \\ \hat{\boldsymbol{y}}(t_1; \boldsymbol{y}(t_0)) - \boldsymbol{y}(t_1) \\ \vdots \\ \hat{\boldsymbol{y}}(t_k; \boldsymbol{y}(t_{K-1})) - \boldsymbol{y}(t_K) \end{bmatrix}, \tag{2.32}$$

the values of the state and costate, $\boldsymbol{y}(t_k)$, $k = 0, \cdots, K$, are determined. The fundamental steps of this method are summarized in Algorithm 2.2.

Algorithm 2.2 Indirect Multiple-shooting method

1: **Input**: Interior points t_k, $k = 0, \cdots, K$, and initial estimate of unknown initial conditions $\boldsymbol{y}(t_0)$ as well as values of states and costates at interior points $\boldsymbol{y}(t_k)$, $k = 1, \cdots, K-1$
2: **Initialization**
3: **for** Each interval $[t_k, t_{k+1}]$, $k = 0, ..., K-1$ **do**
4: Integrate eq. (2.31) to get values of the state and costate $\hat{\boldsymbol{y}}(t_k)$
5: **end for**
6: Main step:
7: **while** $||F_m|| > 0$ the tolerance **do**
8: Update the initial guess
9: **for** Each interval $[t_k, t_{k+1}]$, $k = 0, ..., K-1$ **do**
10: Integrate eq. (2.31) to get values of the state and costate $\hat{\boldsymbol{y}}(t_{k+1})$
11: **end for**
12: **end while**
13: **output**: The optimal trajectory

Note that the multiple shooting approach involves solving more equations and more unknown parameters than the single shooting method. As stated in [15], the exponential growth of errors in the unknown boundary conditions can be minimized by splitting up the domain into subintervals. Multiple shooting might also cause problems if the costate is not estimated with sufficient accuracy.

First-order optimality criteria needs to be satisfied via an indirect method, and deriving these conditions

for complex systems can be challenging. The Hamiltonian system is challenging to solve as it is inherently unstable, even if the optimal control problem yields the first-order optimality criteria. In addition, it is very non-intuitive to have to supply a first estimate for the costate. In the next section, an approach that may be utilized to overcome these challenges is presented.

2.3 Direct Methods

Direct Methods based on control parameterization in a finite dimension have gained prominence due to their ability to address complex control problems in finite-dimensional systems [21]. These methods offer a versatile framework for tackling control optimization tasks, such as trajectory tracking, system stabilization, and optimal control, by parameterizing the control input over a finite time horizon.

The fundamental idea behind this approach is to discretize the control input over a finite time interval, thereby transforming the continuous-time control problem into a finite-dimensional optimization problem. Then a continuous optimal control problem is transcribed into a finite-dimensional nonlinear programming problem (NLP):

$$\text{minimize} \quad \boldsymbol{f}(\boldsymbol{y}) \tag{2.33}$$

$$\text{subject to} \quad \boldsymbol{g}(\boldsymbol{y}) \leq 0 \tag{2.34}$$

$$\boldsymbol{h}(\boldsymbol{y}) = 0. \tag{2.35}$$

where the objective function in (2.4) is represented by the function $\boldsymbol{f}$; the vector of equality constraints in (2.6) and (2.7) is $\boldsymbol{g}$; and the vector of inequality constraints in (2.8)-(2.10) is $\boldsymbol{h}$.

State-of-the-art NLP solvers may be used to solve the resultant NLP [4, 22]. Direct methods include direct shooting, direct multiple shooting, and direct collocation methods; these methods are similar to the strategies employed in indirect methods. This approach provides a systematic way to find optimal or near-optimal control policies by optimizing a set of control parameters, often represented as a vector, subject to system dynamics and constraints.

2.3.1 Direct Shooting and Multiple-Shooting Methods

According to [13], the single shooting method discretizes the trajectory's control $\boldsymbol{u}(t)$ of an Optimal Control Problem. The resulting shooting grid is (see Figure 3)

$$0 = t_0 < t_1 < \cdots < t_N = t_f.$$

Conventional approaches for discretizing $\boldsymbol{u}(t)$ include cubic splines functions, piecewise constant functions, and linear functions. In this work, the piecewise constant function is taken into consideration for simplicity, and, therefore, $\boldsymbol{u}(t)$ is discretized by the parameters $\boldsymbol{q}_k$, using the formula

$$\boldsymbol{u}(t) = \boldsymbol{q}_k \quad \text{for} \quad t \in [t_k, t_{k+1}]$$

with $k = 0, \cdots, N-1$. The control input for each time step k has the size n_q, i.e. $\boldsymbol{q} \in \mathcal{R}^{(N-1)\times n_q}$. As a result, the system states $\boldsymbol{x}$ may be described as variables depending on time t and, when a starting value of $\boldsymbol{x}_0$ is specified, the governing dynamics can be written as follows:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t; \boldsymbol{q})), \quad \forall t \in [t_0, t_N] \tag{2.36}$$

$$\boldsymbol{x}(0) = \boldsymbol{x}_0. \tag{2.37}$$

The symbol $(t;)$ indicates that t is a free variable, and the semicolon is followed by a list of additional parameters. Schemes like the Euler or Heun can be used to numerically integrate $\boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t; \boldsymbol{q}))$.

The single shooting method is shown in Figure 3 for a problem with a single system state $\boldsymbol{x}(t; \boldsymbol{u})$ and a one-dimensional control input $\boldsymbol{u}(t; \boldsymbol{q})$.

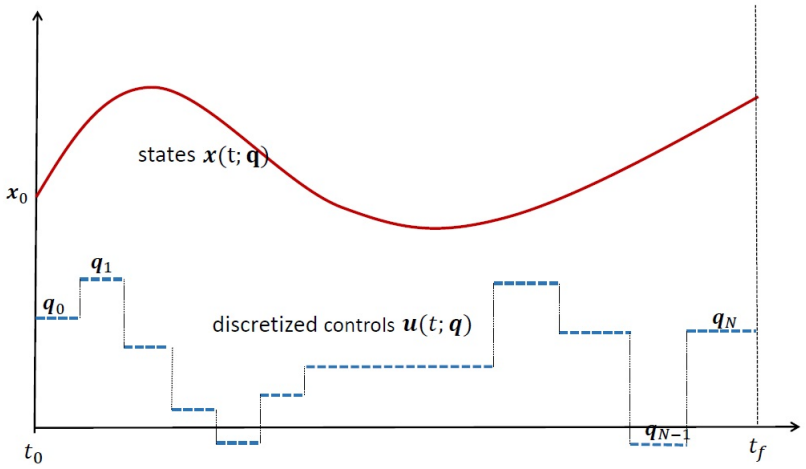

Figure 3: Principle of single shooting

The NLP of an Optimal Control Problem discretized by a single shooting concludes with the following:

$$\min_{\boldsymbol{q}} \Phi(\boldsymbol{x}(t_N;\boldsymbol{q})) + \sum_{k=0}^{N-1} L(t, \boldsymbol{x}(t_k;\boldsymbol{q}), \boldsymbol{q}_k) \tag{2.38}$$

subjected to:

$$\boldsymbol{g}(\boldsymbol{x}(t_k;\boldsymbol{q}), \boldsymbol{q}_k) \leq 0, \quad k = 0, \cdots, N-1 \tag{2.39}$$

$$\Psi(\boldsymbol{x}(t_N;\boldsymbol{q})) = 0. \tag{2.40}$$

Algorithm 2.3 summarizes the steps involved in a direct single shooting approach.

Algorithm 2.3 Direct Single-shooting method

1: **Input**: Give an artificial start estimate for the control trajectory parameterization's parameters.
2: Split the total time periods into N smaller intervals: $t_0 = t_1 < \cdots < t_N = t_f$
3: Using the finite parameters $\boldsymbol{q} = [\boldsymbol{q}_0, \cdots, \boldsymbol{q}_{N\text{-}1}]$ of the chosen suitable polynomials, approximate the control in each time interval.
4: **while** The cost function is not optimal or any constraints are not satisfied **do**
5: After obtaining the finite control parameters in the previous step, use numerical integral techniques to get the state variables.
6: Evaluate the obtained NLP: $\Phi(\boldsymbol{x}(t_f,), t_f) + \int_{t_0}^{t_f} (L(t, \boldsymbol{x}(t), \boldsymbol{u}(t, \boldsymbol{q}))\, dt,$
7: Check the terminal conditions and discretized path constraints.
8: Update the approximation of $\boldsymbol{q}$ with NLP solvers
9: **end while**

For a problem with a single system state, a one-dimensional control input, and a terminal constraint, use the single shooting discretization approach [13]. The comparatively limited optimization variables and limitations are a benefit of single shooting. Single shooting has the benefit of having a limited number of optimization variables and constraints. This method is easy to employ since only the control variables require a preliminary guess. On the other hand, the user is unable to offer previous knowledge of the state trajectory $\boldsymbol{x}(t)$ in single shooting.

Similar to single shooting, the control trajectory $\boldsymbol{u}(t)$ is parameterized by the discrete values $\boldsymbol{q}_k$, which are constant between the discretization points of the shooting grid $0 = t0\ t1\ ...\ tN = T$. It's crucial to note that the N intervals don't have to be equally spaced. The discretization of $\boldsymbol{u}(t)$, for $t \in [t_k, t_{k+1}]$, is given by

$$\boldsymbol{u}(t) = \boldsymbol{q}_k.$$

After parameterizing the state trajectory $\boldsymbol{u}(t)$ over the shooting grid, the N Initial Value Problems are produced. Starting with the artificial initial value χ_k, a set of ordinary differential equations is solved individually on each interval $[t_k, t_{k+1}]$.

$$\dot{\boldsymbol{x}}_k(t) = \boldsymbol{f}(\boldsymbol{x}_k(t), \boldsymbol{q}_k), \quad t \in [t_k, t_{k+1}] \tag{2.41}$$

$$\boldsymbol{x}_k(t_k) = \boldsymbol{\chi}_k. \tag{2.42}$$

To guarantee the continuity of state variables across adjacent subintervals, the following $N-1$ matching specifications are derived

$$\boldsymbol{\chi}_{k+1} = \boldsymbol{x}(t_{k+1}; \boldsymbol{\chi}_k, \boldsymbol{q}_k).$$

The direct multiple shooting discretization concept is illustrated in Figure 4.

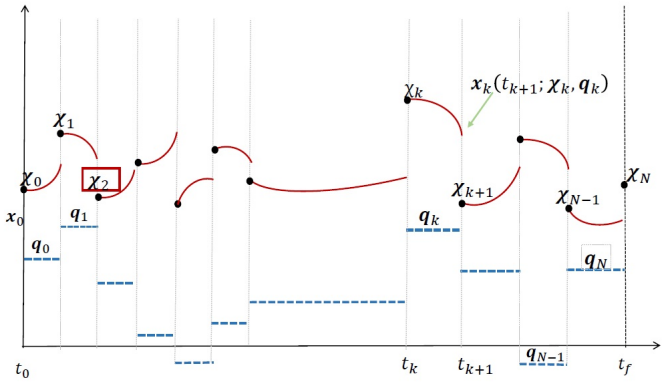

Figure 4: Illustration of multiple shooting

The formulation of the resulting structured NLP reads:

$$\min_{\boldsymbol{\chi}, \boldsymbol{q}} \Phi_N(\boldsymbol{\chi}_N) + \sum_{k=0}^{N-1} L_k(\boldsymbol{\chi}_k, \boldsymbol{q}_k), \boldsymbol{q}_k) \tag{2.43}$$

subjected to

$$\boldsymbol{\chi}_{k+1} = \boldsymbol{x}(t_{k+1}; \boldsymbol{\chi}_k, \boldsymbol{q}_k), \tag{2.44}$$

$$\boldsymbol{\chi}(0) = \boldsymbol{x}_0, \tag{2.45}$$

$$\boldsymbol{g}(\boldsymbol{\chi}_k, \boldsymbol{q}_k) \leq 0, \quad k = 0, \cdots, N-1, \tag{2.46}$$

$$\Psi(\boldsymbol{\chi}_N) = 0. \tag{2.47}$$

The objective function (2.43) sums over the running costs per interval

$$L_k(\boldsymbol{\chi}_k, \boldsymbol{q}_k) = \int_{t_k}^{t_{k+1}} L(t_k; \boldsymbol{x}_k(t_k; \boldsymbol{\chi}_k, \boldsymbol{q}_k), \boldsymbol{q}_k) dt.$$

Additionally, (2.43c) limits the first artificial state vector's starting value to be $\boldsymbol{\chi}_0$ and (2.43b) guarantees continuity between the artificial states $\boldsymbol{\chi}_{k+1}$ and the physical ones $\boldsymbol{x}(\boldsymbol{\chi}_k, \boldsymbol{q}_k)$.

The Direct Multiple-shooting method is summarized in Algorithm 2.4, according to [26]:

Algorithm 2.4 Direct Multiple-shooting method

1: **Input**: Give an artificial start estimate for the control trajectory parameterization's parameters.
2: Split the total time periods into N smaller intervals: $t_0 = t_1 < \cdots < t_N = t_f$
3: Using the finite parameters $\boldsymbol{q} = [\boldsymbol{q}_0, \cdots, \boldsymbol{q}_{N\text{-}1}]$ of the chosen suitable polynomials, approximate the control in each time interval.
4: Using the finite parameters $\boldsymbol{\chi} = [\boldsymbol{\chi}_0, \cdots, \boldsymbol{\chi}_{N\text{-}1}]$ of the chosen suitable polynomials, approximate the state variables in each time interval.
5: **while** The cost function is not optimal or any constraints are not satisfied **do**
6: After obtaining the finite control parameters in the previous step, use numerical integral techniques to get the state variables.
7: Evaluate the obtained NLP: $\Phi(\boldsymbol{x}(t_f, \boldsymbol{\chi}), t_f) + \int_{t_0}^{t_f} (L(t, \boldsymbol{x}(t, \boldsymbol{\chi}), \boldsymbol{u}(t, \boldsymbol{q}))\, dt,$
8: Check the terminal conditions and discretized path constraints.
9: Update the approximation of $\boldsymbol{q}$ and $\boldsymbol{\chi}$ with NLP solvers
10: **end while**

2.3.2 Direct Collocation

In a direct collocation method, both the controls and the states are discretized at a set of mesh points [13] and the dynamic system for $t \in [0, t_f]$

$$\dot{\boldsymbol{x}} - \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t)) = 0 \tag{2.48}$$

is replaced by finitely many equality constraints

$$\boldsymbol{h}_k(\boldsymbol{q}_k, \boldsymbol{\chi}_k, \boldsymbol{\chi}'_k, \boldsymbol{\chi}_{k+1}) = 0. \tag{2.49}$$

where, using the first-order approximations, $\boldsymbol{h}_k$ can be written as follows

$$\boldsymbol{h}_k(\boldsymbol{q}_k, \boldsymbol{\chi}_k, \boldsymbol{\chi}'_k, \boldsymbol{\chi}_{k+1}) = \frac{\boldsymbol{\chi}_{k+1} - \boldsymbol{\chi}_k}{t_{k+1} - t_k} - \boldsymbol{f}\left(\frac{\boldsymbol{\chi}_{k+1} + \boldsymbol{\chi}_k}{2}, \boldsymbol{q}_k\right) = 0. \tag{2.50}$$

The intermediate states between the primary discretization points k are represented by the symbols $\boldsymbol{\chi}'_k$. The final NLP might be expressed as:

$$\min_{\boldsymbol{\chi},\boldsymbol{q}} \Phi_N(\boldsymbol{\chi}_N) + \sum_{k=0}^{N-1} L_k(\boldsymbol{q}_k, \boldsymbol{\chi}_k, \boldsymbol{\chi}_{k+1}) \tag{2.51}$$

subjected to

$$\boldsymbol{h}(\boldsymbol{q}_k, \boldsymbol{\chi}_k, \boldsymbol{\chi}'_k, \boldsymbol{\chi}_{k+1}) = 0, \tag{2.52}$$

$$\boldsymbol{\chi}(0) = \boldsymbol{\chi}_0, \tag{2.53}$$

$$\boldsymbol{g}(\boldsymbol{\chi}_k, \boldsymbol{q}_k) \leq 0, \;\; k = 0, \cdots, N-1, \tag{2.54}$$

$$\psi_f(\boldsymbol{\chi}_N) \tag{2.55}$$

where L_k can approximately combine the operating costs

$$L_k(\boldsymbol{\chi}_k, \boldsymbol{\chi}_{k+1}, \boldsymbol{q}_k) = \int_{t_k}^{t_{k+1}} L(\boldsymbol{x}(t), \boldsymbol{u}(t))dt \approx L\left(\frac{\boldsymbol{\chi}_{k+1} + \boldsymbol{\chi}_k}{2}\right)(t_{k+1} - t_k).$$

The sparseness of the generated NLP is one of the advantages of direct collocation [13]. In contrast to multiple shooting, it has a higher dimensionality. During the iterative solution process, direct collocation exhibits fast local convergence. Additionally, the state trajectory $\boldsymbol{x}(t)$ may be initialized before the optimization process, thus this discretization method can handle systems that are intrinsically unstable [13].

The algorithm 2.5 that follows summarizes the Direct Collocation Method:

Algorithm 2.5 Direct Collocation method

1: **Input**: Give an artificial start estimate for the control trajectory parameterization's parameters.
2: Split the total time periods into N smaller intervals: $t_0 = t_1 < \cdots < t_N = t_f$
3: Construct and provide the initial value (optionally) of the NLP variables with the state, control, static parameters and the terminal time at each node: $\boldsymbol{z} = [\boldsymbol{x}_0, \boldsymbol{u}_0, \boldsymbol{x}_1, \boldsymbol{u}_1, ..., , \boldsymbol{x}_N, \boldsymbol{u}_N, t_f]$
4: **while** The cost function is not optimal or any constraints are not satisfied **do**
5: Evaluate the obtained NLP: $\Phi(\boldsymbol{x}(t_f,), t_f) + \int_{t_0}^{t_f} (L(t, \boldsymbol{x}(t), \boldsymbol{u}(t))\, dt,$
6: Evaluate the discretized constraints.
7: Calculate the gradient of the cost function.
8: Calculate the jacobian matrix of the constraints function.
9: Update the NLP variables $\boldsymbol{z}$
10: **end while**

2.4 Hamilton-Jacobi-Bellman

The Hamilton-Jacobi-Bellman (HJB) partial differential equations (PDE) and Dynamic Programming are foundational concepts in Mathematics as they play a pivotal role in solving complex problems involving optimization and decision-making under uncertainty.

Dynamic programming was first introduced by Richard Bellman and breaks down complex decision processes into smaller and more manageable sub-problems and then optimally combines their solutions to find the best overall decision strategy. This method is particularly useful in situations where decisions are made sequentially, and the consequences of one decision affect future choices.

The Hamilton-Jacobi-Bellman equations are named after William Rowan Hamilton, Carl Gustav Jacob Jacobi and Richard Bellman. These are a type of partial differential equations that characterize the value function of an optimal control problem, which describes the expected cumulative reward or cost in continuous time while making sequential decisions under uncertainty and providing a formalism for solving optimal control problems in continuous-time stochastic systems.

HJB Equation

As a result of the Bellman principle, the Hamilton-Jacobi-Bellman (HJB) equation for optimal control problem (2.5)-(2.10) may be given by:

$$\frac{\partial}{\partial t}V(\boldsymbol{x},t)+\min_{\boldsymbol{u}\in\mathcal{U}}H\left(\boldsymbol{x},\boldsymbol{u},\nabla_{\boldsymbol{x}}V(\boldsymbol{x},t),t\right)=\mathbf{0},\quad t\in(0,t_f), \tag{2.56}$$

$$V(\boldsymbol{x},t_f)=\phi(\boldsymbol{x}),\quad \boldsymbol{x}\in\mathbb{R}^n, \tag{2.57}$$

where H is the Hamiltonian described in (2.12) and the function:

$$V(\boldsymbol{x},t)=\min_{\boldsymbol{u}(\cdot)}J=\int_t^{t_f}L(\boldsymbol{x},\boldsymbol{u},s)ds+\phi(\boldsymbol{x}(t_f),t_f). \tag{2.58}$$

After solving the HJB equation (2.56)-(2.57), the optimal control function can be obtained from:

$$\boldsymbol{u}^*(t)=\min_{\boldsymbol{u}\in\mathcal{U}}H\left(\boldsymbol{x},\boldsymbol{u},\nabla_{\boldsymbol{x}}V(\boldsymbol{x},t),t\right), \tag{2.59}$$

with the state function being the solution of the Ordinary Differential Equations that follows:

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{dt}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}^*, t), \quad \boldsymbol{x}(0) = \boldsymbol{x}_0, t \in [0, t_f]. \tag{2.60}$$

Parallel to Bellman's Equation

The Bellman equation in discrete time is given by:

$$V(\boldsymbol{x}, t) = \min_{\boldsymbol{u}} \left[L(\boldsymbol{x}, \boldsymbol{u}, t) + V((\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t), t + \Delta t) \right] \tag{2.61}$$

In both the HJB equation and the Bellman equation, the property of optimality is exhibited. This parallel highlights that the HJB provides a continuous-time framework for solving optimal control problems, while the Bellman equation in Dynamic Programming addresses discrete-time problems.

Combining both these equations together in an optimal control context allows us to find the optimal control in problems that involve continuous changes over time.

In Bellman's equation (2.61) the $\mathcal{L}(\boldsymbol{x}, \boldsymbol{u}, t)$ represents the immediate cost, $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t)$ represents the transition to the next state, and $V(\mathbf{f}(\boldsymbol{x}, \boldsymbol{u}, t), t + \Delta t)$ represents the value function at the next time step.

Then a transition to continuous-time is made by taking the limit as Δt approaches zero as follows:

$$\lim_{\Delta t \to 0} V(\boldsymbol{x}, t) = \lim_{\Delta t \to 0} \min_{u} \left[L(\boldsymbol{x}, \boldsymbol{u}, t) + V(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t), t + \Delta t) \right]$$

In continuous-time, the minimum control policy if found by minimizing the Hamiltonian $\mathcal{L}(\boldsymbol{x}, \boldsymbol{u}, t, V)$:

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{u}, t, V) = L(\boldsymbol{x}, \boldsymbol{u}, t) + \frac{\partial V}{\partial t}(\boldsymbol{x}, t) + \nabla V(\boldsymbol{x}, t) \cdot \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t)$$

Since the continuous-time Bellman equation is a limiting case of the discrete-time Bellman equation, by substituting the Hamiltonian into the limit equation, the continuous-time Bellman equation can be obtained, which is equivalent to the HJB equation:

$$\lim_{\Delta t \to 0} V(\boldsymbol{x}, t) = \min_{u} \left[\mathcal{L}(\boldsymbol{x}, \boldsymbol{u}, t, V) \right]$$

The theoretical framework described in this chapter will be used in next chapters to solve the numerical lap-time-optimal control problem, using the Orthogonal Collocation Methods, also know as Pseudospectral method.

Chapter 3

Optimal Lap Time for a Racing Vehicle

In this chapter, the kinematics of the track and vehicle are modeled using results from classical differential geometry. The minimum lap time optimal control problem is formulated and, applying the Pontryagin Maximum Principle, the respective Hamiltonian boundary-value problem is established. This Chapter is mostly supported by the work referenced in [7].

3.1 Track model

The track is modelled as in [7], considering the following assumptions:

- **Coordinate system**: Model the route using a curvilinear coordinate system that uses the arc length of the reference line as the abscissa. This system is commonly used for race track modeling.
- **Trajectory description**: The trajectory is described by the curvature of the reference line, called κ, s represents the centerline of the track and N_l and N_r denote the width on the left and right of the reference line, respectively.
- **Vehicle parameters**: The position and orientation of the vehicle on the track are described by the following variables:
  - s: The abscissa of the curve along the reference line.
  - n: Lateral displacement relative to the reference line (if necessary, the displacement occurs on the normal vector of each point).
  - ξ : The relative angle between the tangent vector and the direction of the vehicle.
  - β: Slip angle, which determines the direction of the velocity vector.

At any point s, the curvature at that point can be obtained by inverting the local radius of curvature R. The tangent vector on the reference line t is described by the track orientation angle θ. The location and orientation of the vehicle can be described by the curvilinear abscissa along the reference line s, the lateral displacement from the reference line n and the relative angle ξ between the tangent vector and the vehicle orientation. The direction of the velocity vector depends on the side slip angle β. Figure 5 shows the vehicle body frame with Cartesian coordinates.

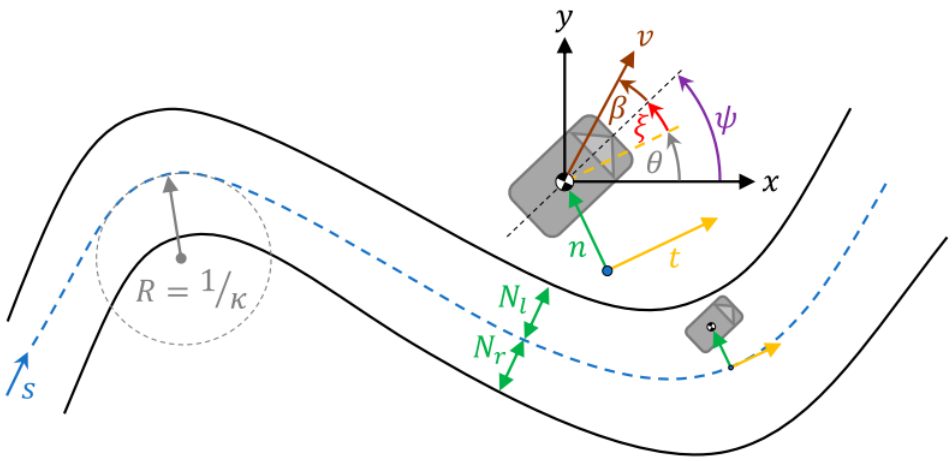

Figure 5: Illustration of a track segment ant the curvilinear coordinate systems - figure adopted from [7].

The equations that describe the evolution over time of the coordinates s, n and ξ are described as:

$$\dot{s} = \frac{v\cos(\xi+\beta)}{1-n\kappa}, \tag{3.1a}$$

$$\dot{n} = v\sin(\xi+\beta), \tag{3.1b}$$

$$\dot{\xi} = \omega_z - \kappa\frac{v\cos(\xi+\beta)}{1-n\kappa}, \tag{3.1c}$$

where ω_z is the yaw rate.

3.2 Kinematic of the vehicle and force model

Vehicles may be viewed as several rigid body systems that include mechanical parts, motion pairs, and other components. The vehicle model is a very complex nonlinear system that is sometimes challenging to adequately represent using one or more mathematical formulas. A simple double-track model proposed by [7] reduces the vehicle position to a single point acting on the center of mass. The model has three degrees of freedom, as presented in Figure 6: the speed v, the sideslip angle β and the yaw rate ψ.

The model also combines a nonlinear tire model, an approximate description of the engine characteristics and a quasi-steady state wheel load transfer.

In Figure 6, the y-axis is lateral direction and positive on the left, the x-axis is longitudinal direction and represents the vehicle's running direction, v represents the velocity at the centre of gravity, l_i is the distance between the axle and α_{ij} represents the centroid for each tire slip angle,. $F_{x,ij}$ and $F_{y,ij}$ are the longitudinal and lateral tires forces, respectively, where the index i represents the front and rear axle, while index j distinguishes between left and right side of the vehicle. Front and rear axle are represented by $i = f, r$, whereas left and right side of the vehicle are represented by $j = l, r$.

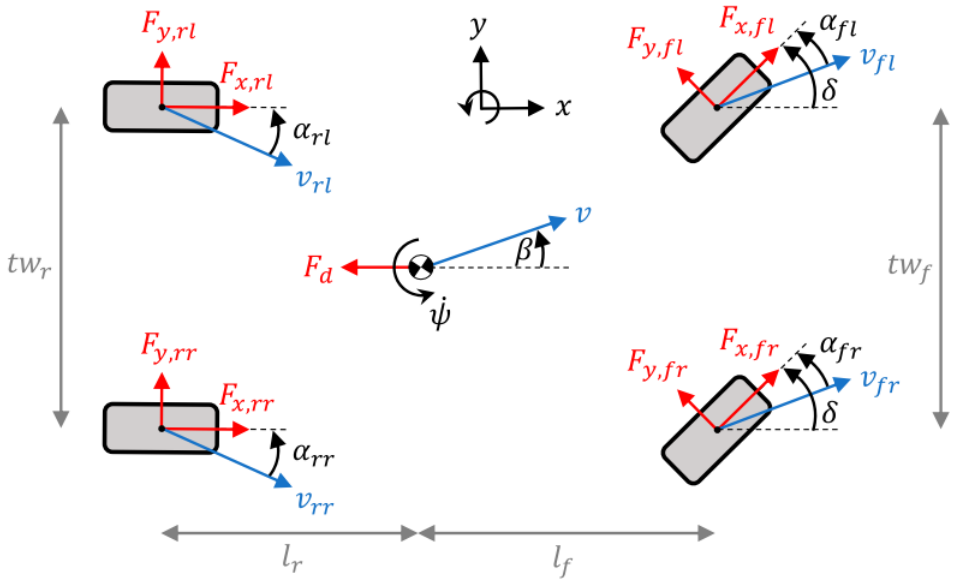

Figure 6: Illustration of wheel model forces diagram (see in [7])
)

The steering wheel angle (angle of front wheel vs general direction of the car's longitudinal axis) is named δ, while the slip angle β gives the angle of the car's direction of motion relative to this axis. Finally, ψ and ω_z represent the car's angle of inclination and its angular velocity, which represents the direction of the car's vertical axis compared to the horizontal coordinate axis.

The variable states are listed in Table 1, while Figure 6 visualizes the choice of coordinates, angles and forces.

State	Unit	State Unit Description
x	m	Horizontal position of the car
y	m	Vertical position of the car
v	$\frac{m}{s}$	Magnitude of directional velocity of the car
δ	rad	Steering wheel angle
β	rad	Side slip angle rad Yaw angle
ψ	rad	Yaw angle velocity

Table 1: Coordinate and state variables used in the model of the car.

The centre of gravity is represented by the coordinate pair (x, y) obtained by the integrated speed direction:

$$\dot{x}(t) = v(t)\cos(\psi(t) - \beta(t)), \tag{3.2}$$

$$\dot{y}(t) = v(t)\sin(\psi(t) - \beta(t)). \tag{3.3}$$

The acceleration is obtained by the total force acting on the mass m of the car in the driving direction:

$$\begin{aligned}\dot{v} =& \frac{1}{m}\Big((F_{x,rl} + F_{x,rr})\cos(\beta) + ((F_{x,fl} + F_{x,fr})\cos(\delta - \beta), \\ &+ (F_{y,rll} + F_{y,rr})\sin(\beta)) - (F_{y,fl} + F_{y,fr})\sin(\delta - \beta)) - \frac{1}{2}c_d\,\rho\,A\,v^2\cos(\beta)\Big),\end{aligned} \tag{3.4}$$

where ρ denotes the air density, A the frontal area of the vehicle and c_d is the aerodynamic drag force coefficient. The last term of the above equation means that the aerodynamic drag force is assumed to be quadratic in the vehicle speed and is applied at the centre of pressure, which is located at the centre of gravity of the vehicle for simplicity.

The controlled angular velocity gives us the steering wheel angle:

$$\dot{\delta}(t) = \omega_z. \tag{3.5}$$

The steering wheel regulates the slip angle β, and is described by the following equation

$$\begin{aligned}\dot{\beta} =& -\omega_z + \frac{1}{mv}\Big(-(F_{x,rl} + F_{x,rr})\sin(\beta) + (F_{x,fl} + F_{x,fr})\sin(\delta - \beta) \\ &+ (F_{y,rl} + F_{y,rr})\cos(\beta) + (F_{y,fl} + F_{y,fr})\cos(\delta - \beta) + \frac{1}{2}c_d\rho\,A\,v^2\sin(\beta)\Big).\end{aligned} \tag{3.6}$$

The yaw angle ψ is calculated by integrating over its change, ω_z, as

$$\dot{\psi}(t) = \omega_z(t), \tag{3.7}$$

which is the total of forces acting on the front wheel perpendicular to the longitudinal axis of orientation of the vehicle

$$\begin{aligned}\dot{\omega}_z =& \frac{1}{J_{zz}}\Big((F_{x,rr} - F_{x,rl})\frac{tw_r}{2} - (F_{y,rl} + F_{y,rr})l_r \\ &+ ((F_{x,fr} - F_{x,fl})\cos(\delta) + (F_{y,fl} - F_{y,fr})\sin(\delta))\frac{tw_f}{2} \\ &+ ((F_{y,fl} + F_{y,fr})\cos(\delta) + (F_{x,fl} + F_{x,fr})\sin(\delta))l_f\Big),\end{aligned} \tag{3.8}$$

where J_{zz} represents the mass moment of inertia in the vertical axis, tw_f describes the track's width at the front axle and tw_r describes the track's width at the rear axle. The tire lateral forces are given by a linear tire model, while longitudinal ones are given by:

$$F_{x,fj} = \frac{1}{2}k_{drive}F_{drive} + \frac{1}{2}k_{brake}F_{brake} - \frac{1}{2}f_r mg\frac{l_r}{l}, \tag{3.9a}$$

$$F_{x,rj} = \frac{1}{2}(1 - k_{drive})F_{drive} + \frac{1}{2}(1 - k_{brake})F_{brake} - \frac{1}{2}f_r mg\frac{l_f}{l}, \tag{3.9b}$$

where the last term in each equation depicts the rolling resistance, which is calculated with a static rolling resistance f_r and the gravitational acceleration g. The first two terms mean that the braking forces are distributed between the front and the rear axles via the static brake force distribution k_{brake}, while k_{drive} distributes the driving force.

The steering angle δ and the longitudinal force are split into driving force $F_{drive} \geq 0$ and a braking force $F_{brake} \leq 0$ and are selected as control variables.

The interaction between the moving vehicle and the road surface is presented by the tire-road contact forces, which are crucial factors. These forces, however, are nonlinear and dependent on a variety of parameters, such as the quality of the tire, the slip ratio, the side slip angle, etc. Based on a semi-empirical tire model, Pacejka has suggested the tire formula, also known as the magic formula. Pacejka's *Magic Formula* is used to determine the side (lateral) force on the front wheel as a function of the front slip angle α_f,

$$F_y = \mu F_z \sin(C \arctan(B\alpha - E(B\alpha - \arctan(B\alpha)))),$$

in which α is the slip angle, μ represents the road friction coefficient, F_z denotes the tire load F_z, B is

stiffness factor, C is the shape factor and E is the curvature factor, determining the curve shape.

The simple model over-estimates the transferable lateral tire forces, F_y, of the tire, especially at high normal forces F_z. To avoid high lateral tire forces F_y, the tire model is modeled by doing the following:

$$F_{y,ij} = \mu_{ij} F_{z,ij} \left(1 + \epsilon_i \frac{F_{z,ij}}{F_{z,0}}\right) \sin(C_i \arctan(B_i \alpha_{ij} - E_i(B_i \alpha_{ij} - \arctan(B_i \alpha_{ij})))), \quad (3.10)$$

where μ_{ij} is the variable wheel-specific friction coefficient along the racetrack. The ϵ coefficient, satisfying $\epsilon \leq 0$, is used as a degression of tire behaviour against the wheel load.

To complete the Pacejka's tire model, front and rear slip angle must be determined as

$$\alpha_{fl} = \delta - \arctan\left(\frac{l_f \omega_z + v \sin(\beta)}{v \cos(\beta) - \frac{1}{2} t w_f \omega_z}\right), \quad (3.11a)$$

$$\alpha_{fr} = \delta - \arctan\left(\frac{l_f \omega_z + v \sin(\beta)}{v \cos(\beta) + \frac{1}{2} t w_f \omega_z}\right), \quad (3.11b)$$

$$\alpha_{rl} = \arctan\left(\frac{l_r \omega_z - v \sin(\beta)}{v\, cos(\beta) - \frac{1}{2} t w_r \omega_z}\right), \quad (3.11c)$$

$$\alpha_{rr} = \arctan\left(\frac{l_r \omega_z - v \sin(\beta)}{v\, cos(\beta) + \frac{1}{2} t w_r \omega_z}\right). \quad (3.11d)$$

The vertical tire load $F_{z.ij}$ consists of static and dynamic terms, and is approximated by:

$$F_{z,fl} = \frac{1}{2} mg \frac{l_r}{l_f + l_r} - \frac{1}{2} \frac{h_{cog}}{l_f + l_r} m a_x - k_{roll} \Gamma_y + \frac{1}{4} c_{l,f} \rho A v^2, \quad (3.12a)$$

$$F_{z,fr} = \frac{1}{2} mg \frac{l_r}{l_f + l_r} - \frac{1}{2} \frac{h_{cog}}{l_f + l_r} m a_x + k_{roll} \Gamma_y + \frac{1}{4} c_{l,f} \rho A v^2, \quad (3.12b)$$

$$F_{z,rl} = \frac{1}{2} mg \frac{l_f}{l_f + l_r} + \frac{1}{2} \frac{h_{cog}}{l_f + l_r} m a_x - (1 - k_{roll}) \Gamma_y + \frac{1}{4} c_{l,r} \rho A v^2, \quad (3.12c)$$

$$F_{z,rr} = \frac{1}{2} mg \frac{l_f}{l_f + l_r} + \frac{1}{2} \frac{h_{cog}}{l_f + l_r} m a_x + (1 - k_{roll}) \Gamma_y + \frac{1}{4} c_{l,r} \rho A v^2, \quad (3.12d)$$

where a_x is the longitudinal acceleration at the centre of gravity, a_y represents the lateral acceleration at the centre of gravity, h_{cog} denotes the height of the centre of gravity, Γ_y is the lateral wheel load transfer, $c_{l,f}$ represents the lift coefficient for the front axle and $c_{l,r}$ is the lift coefficient for the rear axle.

The dynamic wheel load change Γ_y is calculated using the current lateral acceleration a_y and the geometric dimensions of the vehicle:

$$\Gamma_y = \frac{h_{cog}}{\frac{1}{2}(tw_f + tw_r)} (F_{y,rl} + F_{y,rr} + (F_{x,fl} + F_{x,fr}) \sin(\delta) + (F_{y,fl} + F_{y,fr}) \cos(\delta)). \quad (3.13)$$

By approximating the longitudinal acceleration a_x for the steering angle δ = 0, one can focus on optimizing the steering angle for lateral control while assuming a certain longitudinal acceleration profile. This makes it so that the value ma_x can be specified as a function of the state and control variables:

$$ma_x \approx F_{drive} + F_{brake} - \frac{1}{2}c_d A v^2 - f_r mg. \tag{3.14}$$

Coulomb's friction circle is seen in Figure 7. According to the Kamm circle, the tire's longitudinal and lateral forces are vector sums that are less than or equal to the friction coefficient of the road surface

$$\left(\frac{F_{x,ij}}{\mu_{ij} \cdot F_{z,ij}}\right)^2 + \left(\frac{F_{y,ij}}{\mu_{ij} \cdot F_{z,ij}}\right)^2 \leq 1 \tag{3.15}$$

where μ_{ij} is the friction coefficient for each of the four wheels along the racetrack.

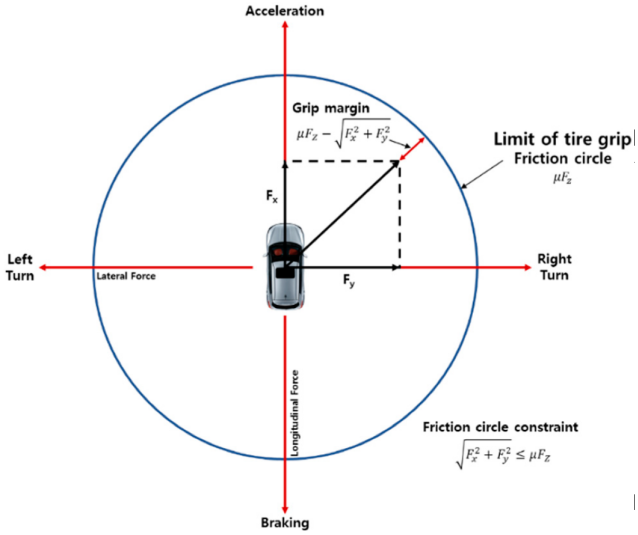

Figure 7: Kamm's circle in [19].

The friction circle, which is stated as follows, should not be exceeded by the driving or braking forces in order for the vehicle to conduct stable driving, i.e.,

$$F_{x_{max}} = \sqrt{(\mu_{ij} F_{z,ij})^2 + (F_{y,ij})^2}. \tag{3.16}$$

Since the problem at hand is an electric motor without gear shifting, the engine map can be approximated by introducing two inequality constraints for the driving force F_{drive} and the power $v \cdot F_{drive}$:

$$v \cdot F_{drive} \leq P_{max}, \tag{3.17a}$$

$$F_{drive} \leq F_{drive,max}. \tag{3.17b}$$

The splitting of the longitudinal force into a driving force F_{drive} and a braking force F_{brake} requires the introduction of the quality constraint:

$$F_{drive} \cdot F_{brake} = 0N^2, \tag{3.18}$$

in order to prevent brake and drive from being applied simultaneously. The steering angle δ is also limited. The maximum available steering angle range can be specified by the inequality constraint:

$$-\delta_{max} \leq \delta \leq \delta_{max}, \tag{3.19}$$

where δ_{max} corresponds to half of the maximum steering angle range available.

In addition to compliance with the maximum permissible values for the control variable, the actuator dynamics of the vehicle must also be taken into account. Detailed modelling of the steering and powertrain dynamics is dispensed in order to maintain a lower model complexity. Instead, only the rates of change of the control variables are limited. The dynamic limitations can be introduced by formulating inequality constraints as follows:

$$\frac{F_{drive,k+1} - F_{drive,k}}{SF_k \cdot \Delta s} \leq \frac{F_{drive,max}}{T_{drive}}, \tag{3.20a}$$

$$\frac{F_{brake,k+1} - F_{brake,k}}{SF_k \cdot \Delta s} \geq \frac{F_{brake,max}}{T_{brake}}, \tag{3.20b}$$

$$\frac{-\delta_{max}}{T_\delta} \leq \frac{\delta_{k+1} - \delta_k}{SF_k \cdot \Delta s} \leq \frac{\delta_{max}}{T_\delta}. \tag{3.20c}$$

3.3 Optimal Control Problem Formulation

The state vector of the Ordinary Differential Equations system are denoted by $\boldsymbol{x}$ and the corresponding righthand side function by $\boldsymbol{f}$ in the dynamic model. In the simple model a possible set of state variables

and controls is:

$$\boldsymbol{x} = (n, \xi, v, \beta, \omega_z)^T, \quad \boldsymbol{u} = \left(\delta, F_{drive}, F_{brake}, \Gamma_y\right)^T$$

all the state variables $\boldsymbol{x}$ derive from the integration of the state equations $\dot{\boldsymbol{x}}$, or rather equations (3.4), (3.6) and (3.8).

The target is to determine the controls $\boldsymbol{u}$ that minimize the *cost functional* J:

$$J = \int_{t_0}^{t_f} 1 \, dt.$$

The equations of motion from the track model and the vehicle model are combined to form the state equations of the Optimal Control Problem for motion planning. The complete model employed in this work consists of

$$\dot{\boldsymbol{x}}(t) = \left[\dot{s}, \dot{n}, \dot{\xi}, \dot{v}, \dot{\beta}, \dot{\omega_z}\right]^T \tag{3.21}$$

where

$$\begin{aligned}
\dot{s} &= \frac{v\cos(\xi+\beta)}{1-n\kappa}, \\
\dot{n} &= v\sin(\xi+\beta), \\
\dot{\xi} &= \omega_z - \kappa\frac{v\cos(\xi+\beta)}{1-n\kappa}, \\
\dot{v} &= \frac{1}{m}\Big((F_{x,rl}+F_{x,rr})\cos(\beta) + (F_{x,fl}+F_{x,fr})\cos(\delta-\beta) \\
&\quad + (F_{y,rll}+F_{y,rr})\sin(\beta) - (F_{y,fl}+F_{y,fr})\sin(\delta-\beta)) - \frac{1}{2}c_d\,\rho\,A\,v^2\cos(\beta)\Big), \\
\dot{\beta} &= -\omega_z + \frac{1}{mv}\Big(-(F_{x,rl}+F_{x,rr})\sin(\beta) + (F_{x,fl}+F_{x,fr})\sin(\delta-\beta) + (F_{y,rl}+F_{y,rr})\cos(\beta) \\
&\quad + (F_{y,fl}+F_{y,fr})\cos(\delta-\beta) + \frac{1}{2}c_d\rho\,A\,v^2\sin(\beta)\Big), \\
\dot{\omega_z} &= \frac{1}{J_{zz}}\Big((F_{x,rr}-F_{x,rl})\frac{tw_r}{2} - (F_{y,rl}+F_{y,rr})l_r + ((F_{x,fr}-F_{x,fl})\cos(\delta)+ \\
&\quad (F_{y,fl}-F_{y,fr})\sin(\delta))\frac{tw_f}{2} + ((F_{y,fl}+F_{y,fr})\cos(\delta) + (F_{x,fl}+F_{x,fr})\sin(\delta))l_f\Big)
\end{aligned}$$

where all model's states and control inputs are a function of time, but for simplicity, this notation is omitted.

The minimum lap time problem involves finding vehicle control inputs that minimize the time t_f required to move the vehicle along the track from the starting line to the finish line. Therefore, the coordinates of the curve s changes between the fixed start point and the end point, while the final value t_f

of the independent variable t is unknown. For this reason, it is better to make a change of the independent variable from t to s in the equations of motion. Such a variable change is based on the following derivative rule:

$$S_f = \frac{dt}{ds} = \left(\frac{ds}{dt}\right)^{-1} = \frac{1 - nk}{v\cos(\xi + \beta)}. \tag{3.22}$$

The quantity S_f is the reciprocal of the component of the velocity of the vehicle in the track-segment direction (on the spine at s). It is feasible to see that, in general,

$$\frac{d\boldsymbol{x}}{ds} = \frac{d\boldsymbol{x}}{dt}\frac{dt}{ds} = S_f\frac{d\boldsymbol{x}}{dt} = S_f \times \dot{\boldsymbol{x}},$$

and, therefore, from (3.1b), (3.1c), (3.4), (3.6) and (3.8), follows:

$$\boldsymbol{x}'(t) = \left[n', \xi', v', \beta', \omega_z'\right]^T \tag{3.23}$$

where

$$\begin{aligned}
n' &= S_f\left[v\sin(\xi + \beta)\right], \\
\xi' &= \omega_z - \kappa\frac{v\cos(\xi + \beta)}{1 - n\kappa}, \\
v' &= \frac{1}{m}\Big((F_{x,rl} + F_{x,rr})\cos(\beta) + (F_{x,fl} + F_{x,fr})\cos(\delta - \beta) + (F_{y,rll} + F_{y,rr})\sin(\beta) \\
&\quad - (F_{y,fl} + F_{y,fr})\sin(\delta - \beta)) - \frac{1}{2}c_d\,\rho\,A\,v^2\cos(\beta)\Big), \\
\beta' &= -\omega_z + \frac{1}{mv}\Big(- (F_{x,rl} + F_{x,rr})\sin(\beta) + (F_{x,fl} + F_{x,fr})\sin(\delta - \beta) + (F_{y,rl} + F_{y,rr})\cos(\beta) \\
&\quad + (F_{y,fl} + F_{y,fr})\cos(\delta - \beta) + \frac{1}{2}c_d\rho\,A\,v^2\sin(\beta)\Big), \\
\omega_z' &= \frac{1}{J_{zz}}\Big((F_{x,rr} - F_{x,rl})\frac{tw_r}{2} - (F_{y,rl} + F_{y,rr})l_r + ((F_{x,fr} - F_{x,fl})\cos(\delta) + \\
&\quad (F_{y,fl} - F_{y,fr})\sin(\delta))\frac{tw_f}{2}((F_{y,fl} + F_{y,fr})\cos(\delta) + (F_{x,fl} + F_{x,fr})\sin(\delta))l_f\Big)
\end{aligned}$$

Here, the prime ($'$) is used to denote the derivative with regard to the traveled distance s. The objective function in the Lagrange form can also be rewritten to describe it as a function of the distance traveled s rather than time t:

$$J = \int_{t_0}^{t_f} dt = \int_{s_0}^{s_f}\frac{dt}{ds}ds = \int_{s_0}^{s_f}\frac{1 - nk}{v\cos(\xi + \beta)}ds,$$

where s_0 is the start point of the track and s_f is the end position.

The final optimal control problem is reformulated by the following problem:

Problem 1. (Minimum-Lap-Time Control)

The minimum-lap-time design and control strategies are the solution of

$$\min_{u} \quad \int_{s_0}^{s_f} \frac{1-nk}{v\cos(\xi+\beta)}\,ds \tag{3.24a}$$

$$\text{s.t. } (3.23), \tag{3.24b}$$

$$(3.13), (3.18), \tag{3.24c}$$

$$(3.15), (3.17a), (3.19c), (3.20). \tag{3.24d}$$

The start line and the finish line, i.e., s_0 and s_f have to be the same in order for the reference line to be closed, since the end of the next lap is equivalent to the beginning of the lap that the vehicle is doing.

Pontryagin Maximum Principle

Now, in order to acquire the necessary conditions for optimality, also known as the Pontryagin Maximum Principle, the Hamiltonian function is defined. Let $\boldsymbol{\lambda}(t) = (\lambda_1(t), \lambda_2(t), \lambda_3(t), \lambda_4(t), \lambda_5(t))^T$ and $\lambda_0 \in \mathbb{R}$. Then, the Hamiltonian for the Optimal Control Problem is given by:

$$\begin{aligned}
H(\lambda_0\,\boldsymbol{\lambda}, \boldsymbol{x}) &= \lambda_0 \frac{1-nk}{v\cos(\xi+\beta)} + \langle \lambda, f(\boldsymbol{x}, \boldsymbol{u})\rangle \\
&= \lambda_0 \frac{1-nk}{v\cos(\xi+\beta)} + \lambda_1(v\sin(\xi+\beta)) + \lambda_2\left(\omega_z - \kappa\frac{v\cos(\xi+\beta)}{1-n\kappa}\right) \\
&+ \lambda_3\Big[\frac{1}{m}\Big((F_{x,rl}+F_{x,rr})\cos(\beta) + (F_{x,fl}+F_{x,fr})\cos(\delta-\beta) \\
&+ (F_{y,rll}+F_{y,rr})\sin(\beta) - (F_{y,fl}+F_{y,fr})\sin(\delta-\beta)) - \frac{1}{2}c_d\,\rho\,A\,v^2\cos(\beta)\Big)\Big] \\
&+ \lambda_4\Big[-\omega_z + \frac{1}{mv}\Big(-(F_{x,rl}+F_{x,rr})\sin(\beta) + (F_{x,fl}+F_{x,fr})\sin(\delta-\beta) \\
&+ (F_{y,rl}+F_{y,rr})\cos(\beta) + (F_{y,fl}+F_{y,fr})\cos(\delta-\beta) + \frac{1}{2}c_d\rho\,A\,v^2\sin(\beta)\Big)\Big] \\
&+ \lambda_5\Big[\frac{1}{J_{zz}}\Big((F_{x,rr}-F_{x,rl})\frac{tw_r}{2} - (F_{y,rl}+F_{y,rr})l_r + ((F_{x,fr}-F_{x,fl})\cos(\delta) \\
&+ (F_{y,fl}-F_{y,fr})\sin(\delta))\frac{tw_f}{2} + ((F_{y,fl}+F_{y,fr})\cos(\delta) + (F_{x,fl}+F_{x,fr})\sin(\delta))l_f\Big)\Big].
\end{aligned} \tag{3.25}$$

According to the Pontryagin Maximum Principle, an extremal for the Optimal Control Problem is a pair $(\boldsymbol{u}^\star, \boldsymbol{x}^\star)$ for which there exists a multiplier $\lambda_0 \geq 0$ and an adjoint vector $\boldsymbol{\lambda}(t)$ satisfying the following

adjoint systems:

$$\lambda_1' = -\frac{\partial H}{\partial n} = \frac{\kappa}{v\cos(\xi+\beta)}\lambda_0 - \frac{v\cos(\xi+\beta)\kappa^2}{(1-n\kappa)^2}\lambda_2, \tag{3.26}$$

$$\lambda_2' = -\frac{\partial H}{\partial \xi} = -\frac{(1-n\kappa)\sin(\xi+\beta)}{v^2\cos^2(\xi+\beta)}\lambda_0 - v\cos(\xi+\beta)\lambda_1 - \kappa\frac{v\sin(\xi+\beta)}{1-n\kappa}\lambda_2, \tag{3.27}$$

$$\lambda_3' = -\frac{\partial H}{\partial v} = \frac{1-nk}{v^2\cos(\xi+\beta)}\lambda_0 - \sin(\xi+\beta)\lambda_1 + \kappa\frac{v\cos(\xi+\beta)}{1-n\kappa}\lambda_2 \tag{3.28}$$

$$\lambda_4' = -\frac{\partial H}{\partial \beta} = -\frac{(1-nk)\sin(\xi+\beta)}{v^2\cos^2(\xi+\beta)}\lambda_0 - v\sin(\xi+\beta)\lambda_1 - \kappa\frac{v\sin(\xi+\beta)}{1-n\kappa}\lambda_2 \tag{3.29}$$

$$\begin{aligned}
&- \lambda_3\Big[\frac{1}{m}\Big(- (F_{x,rl}+F_{x,rr})\sin(\beta) + (F_{x,fl}+F_{x,fr})\sin(\delta-\beta) \\
&+ (F_{y,rll}+F_{y,rr})\cos(\beta) + (F_{y,fl}+F_{y,fr})\cos\delta-\beta)) + \frac{1}{2}c_d\,\rho\,A\,v^2\sin(\beta)\Big)\Big] \\
&- \lambda_4\Big[\frac{1}{mv}\Big(- (F_{x,rl}+F_{x,rr})\cos(\beta) - (F_{x,fl}+F_{x,fr})\cos\delta-\beta) \\
&- (F_{y,rl}+F_{y,rr})\sin(\beta) + (F_{y,fl}+F_{y,fr})\sin(\delta-\beta) + \frac{1}{2}c_d\rho\,A\,v^2\cos(\beta)\Big)\Big]
\end{aligned} \tag{3.30}$$

$$\lambda_5' = -\frac{\partial H}{\partial \omega_z} = -\lambda_2 + \lambda_4 \tag{3.31}$$

and terminal conditions:

$$\lambda_1(t_f) = \lambda_2(t_f) = \lambda_3(t_f) = \lambda_4(t_f) = \lambda_5(t_f) = 0.$$

The Pontryagin Maximum Principle also requires the following additional conditions:

- λ_0 and $\boldsymbol{\lambda}(t)$ do not vanish simultaneously for an optimal control $\boldsymbol{u}^\star$ and its corresponding optimal state $\boldsymbol{x}^\star$;
- The Hamiltonian function H along the optimal solution is constant and has a minimal value.

In the next chapters, the optimal control problem (3.24) will be transcribed into an NLP problem, by applying the direct transcription collocation method. Therefore, the boundary value problem, given by the equations (3.13), (3.26) and (3.31), will not be further considered.

Chapter 4

Pseudospectral Collocation Methods

A Pseudospectral method, also known as discrete variable representation method, is a numerical technique used to approximate solutions to differential equations, and in particular, used to approximate solutions in the context of optimal control problems. The pseudospectral method combines elements of spectral methods together with collocation methods to provide high accurate and efficient solutions.

In the particular case of optimal control problems, the goal is to find the control inputs that minimize an objective function subject to a set of constraints and dynamics described by differential equations. Pseudospectral methods offer a way to discretize these problems and approximate the optimal control inputs. The key idea behind a pseudospectral method is to represent the solution as a weighted sum of basis functions, typically chosen as orthogonal polynomials such as Legendre or Chebyshev polynomials. These basis functions form a spectral basis, which allows for accurate approximation for the solution using a small amount of terms. The method proceeds to complement the spectral basis by an additional pseudospectral basis, which allows the representation of functions on a set of grid points.

4.1 Collocation Points

In a collocation-based pseudospectral method, the differential equations are satisfied at specific points, called collocation points that are within the domain between -1 and 1, by evaluating the differential equations at these points and forming a set of algebraic equations. The collocation points are often chosen as the roots of the basis functions, as this leads to better accuracy. Once the algebraic equations are formed, various techniques can be used to solve the resulting system and obtain the control inputs that satisfy the dynamics and constraints of the problem. The information presented here is mainly based on Masouleh [16], Garg et al. [9], Fahroo and Ross [8], Sowell and Taheri [23], Massaro and Limebeer

[17]. As it was already mentioned, there are many pseudospectral collocation methods, but not all are both suitable for optimal control problems, and accurate as the rest. Having these considerations in mind, out of all pseudospectral collocations methods suited for solving an optimal control problem, the methods chosen were based on having good to very good accuracy and computational efficiency (this is extremely important since the track and vehicle dynamics constraints are considerable). Therefore, the methods that were chosen (all based on Legendre polynomials) are the ones that follow:

- Legendre-Gauss (LG)
- Legendre-Gauss-Lobatto (LGL)
- Legendre-Gauss-Radau (LGR)

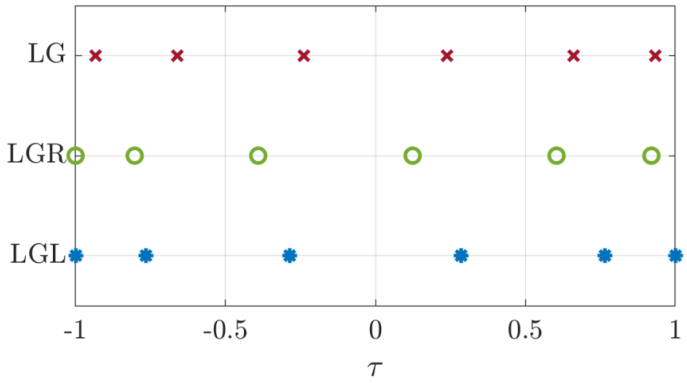

Figure 8: LG, LGR and LGL Collocation points in [23]

When mesh points are specified by $\tau_i \in (-1, 1)$, the mesh points are Gauss points, whereas for Gauss-Lobatto points, $\tau_i \in [-1, 1]$ contains the two boundary points (see Figure 8) , and for Gauss-Radau points, $\tau_i \in (-1, 1]$ or $\tau_i \in [-1, 1)$ contains one of the two boundary points. The LG, LGR and LGL points have the property that:

$$\int_{-1}^{1} p(x)dt \approx \sum_{i=1}^{N} w_i p(\tau_i)$$

is exact for polynomials p of degree at most $2N-1$ (LG), $2N-2$ (LGR) and $2N-3$ (LGL), respectively. Here, w_i denotes the quadrature weights and τ_i the corresponding evaluation nodes. These weights and nodes are selected in Gaussian quadrature such that the quadrature error:

$$E(f) = \int_{-1}^{1} p(x)dt - \sum_{i=1}^{N} w_i p(\tau_i)$$

remains zero for all higher degree polynomials.

The roots of the Nth-degree Legendre polynomial $P_N(\tau)$ are the points called Legendre-Gauss points, with:

$$P_N(\tau) = \frac{1}{2^N n!}\frac{d^N}{dt^N}\left[(\tau^2-1)^N\right]$$

and, therefore, according to [16], the quadrature weights are given by:

$$w_i^{LG} = \frac{2}{(1-\tau_i)^2[P'_N(\tau_i)]^2}. \tag{4.1}$$

Using the condition (4.1), the Gauss pseudospectral differentiation matrix may be computed using the formula:

$$D_{ij}^{LG}(\tau_i) = \begin{cases} \frac{\dot{P}_{N-1}(\tau_i)}{(\tau_i-\tau_j)\dot{P}_{N-1}(\tau_j)}, & i \neq j, 0 \leq i,j \leq N-1 \\ \frac{\tau_i}{1-\tau_j^2}, & i = j \end{cases}$$

Next, τ_0 and τ_{N+1} are defined as non-collocated discretization points, respectively.

It should be noted note that P'_N is the derivative of P_N with respect to τ. The Legendre-Gauss-Radau points are the roots of $P_{N-1}(\tau) + P_N(\tau)$ and, therefore, the weights are expressed as follows:

$$\begin{aligned} w_1^{LGR} &= \tfrac{2}{N^2}, \\ w_i^{LGR} &= \tfrac{2}{(1-\tau_i)[P'_{N-1}(\tau_i)]^2}, \quad i = 2,\cdots,N. \end{aligned} \tag{4.2}$$

Using the condition (4.2), the Gauss pseudospectral differentiation matrix may be computed using the formula

$$D_{ij}^{LGR}(\tau_i) = \begin{cases} -\frac{(N-1)(N-1)}{4}, & i = j = N-1 \\ \frac{P_{N-1}(\tau_i)(1-\tau_j)}{(\tau_i-\tau_j)(1-\tau_j)P_{N-1}(\tau_j)}, & i \neq j, 0 \leq i,j \leq N-1 \\ \frac{1}{2(1-\tau_i)}, & 1 \leq i = j \leq N-1. \end{cases} \tag{4.3}$$

Legendre-Gauss-Lobatto is only accurate up to polynomial order $2N-3$, as two degrees of freedom are lost owing to the border fixed points.

The zeros of P'_{N-1}, where P'_{N-1} is the derivative of the degree $N-1$ Legendre polynomial, make up the inner Legendre-Gauss-Lobatto nodes. The weights for the Legendre-Gauss-Lobatto quadrature become:

$$\begin{aligned} w_{1,N}^{LGL} &= \tfrac{2}{N(N-1)}, \\ w_i^{LGL} &= \tfrac{2}{N(N-1)[P_{N-1}(\tau_i)]^2}, \quad i = 2,\cdots,N-1. \end{aligned} \tag{4.4}$$

Using the condition (4.4), the Gauss pseudospectral differentiation matrix may be computed using the

formula from [23]:

$$D_{ij}^{LGL}(\tau_i) = \begin{cases} -\frac{N(N-1)}{4}, & i = j = 0 \\ \frac{P_{N-1}(\tau_i)}{(\tau_i - \tau_j)P_{N-1}(\tau_j)}, & i \neq j, 0 \leq i, j \leq N-1 \\ 0, & 2 \leq i = j \leq N-2 \\ \frac{N(N-1)}{4}, & i = j = N-1. \end{cases}$$

Since pseudospectral quadrature techniques are usually described on the interval $[-1, +1]$. The fixed interval is divided into N subintervals, with each segment represented by $\mathcal{S}_k = [\tau_{k-1}, \tau_k]$, $k = 1, \cdots, N$, where

$$-1 = \tau_0 < \tau_1 < \cdots < \tau_N = +1$$

are the mesh points. Let $\{\ell_1(\tau), \ell_2(\tau), \cdots, \ell_N(\tau)\}$ be a set of Lagrange basis polynomials given by:

$$\ell_j(\tau) = \prod_{i=k,\ j \neq i}^{N} \frac{(\tau - \tau_i)}{(\tau_j - \tau_i)}, \quad k \leq j \leq N, \tag{4.5}$$

where $k = 1$ if the initial point is collocated; otherwise, $k = 0$ if is not a collocation point. $\ell_j(\tau)$ is the Lagrangian *basis* polynomial with the dimensionless time $\tau \in [0, 1]$. Note that the Lagrange basis polynomials $\ell_j(\tau)$ satisfy the following property:

$$\ell_j(\tau_i) = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$$

This connection may be rephrased using the Kronecker delta as $\ell_j(\tau_i) = \delta_{ij}$. The k-th component of the state is then interpolated:

$$x_k(\tau) \approx \sum_{j=0}^{N} x_{jk} \ell_j(\tau), \quad k = 1, \cdots, n, \tag{4.6}$$

where x_{jk} is the value of the *k*-th component of the state at the *j*-th collocation point and satisfy the condition:

$$x_{jk} = x_k(\tau_j),$$

with $j = 0, \cdots, N$ and $k = 1, ..., n$. After differentiating (4.2)) with regard to τ and evaluating the

result at the *i*-th collocation point, τ_i, one get:

$$\dot{x}_k(\tau_i) \approx \sum_{j=1}^{N} x_{jk}\dot{\ell}_j(\tau_i) = \sum_{j=1}^{N} x_{jk}D_{ij}, \quad D_{ij} = \dot{\ell}_j(\tau_i). \tag{4.7}$$

It is significant to observe that the differentiation matrix has order $N+1$ by N since the series (4.2) includes Lagrange polynomial $\ell_0(\tau)$ associated with the non-collocated point $\tau_0 = -1$. In matrix form, equation (4.7) becomes:

$$\dot{x}_k(\tau_j) = [\boldsymbol{D}\boldsymbol{X}]_{ik}$$

where the Differentiation Matrices, $\boldsymbol{D} = (D_{ij}) \in \mathbb{R}^{N\times(N+1)}$, and state vector, $\boldsymbol{X} \in \mathbb{R}^{(N+1)\times n}$, are expressed as:

$$\boldsymbol{D} = \begin{bmatrix} \dot{\ell}_0(\tau_1) & ,\dot{\ell}_1(\tau_1) & \dot{\ell}_2(\tau_1) & \cdots & \dot{\ell}_N(\tau_1) \\ \dot{\ell}_0(\tau_2) & \dot{\ell}_1(\tau_2) & \dot{\ell}_2(\tau_2) & \cdots & \dot{\ell}_N(\tau_2) \\ \vdots & \vdots & \ddots & \vdots & \\ & & & & \\ \dot{\ell}_0(\tau_N) & \dot{\ell}_1(\tau_N) & \dot{\ell}_2(\tau_N) & \cdots & \dot{\ell}_N(\tau_N) \end{bmatrix}, \quad \boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}_0 \\ \boldsymbol{X}^{CP} \end{bmatrix}, \boldsymbol{X}^{CP} = \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_N \end{bmatrix}$$

where $\boldsymbol{x}_i \in \mathbb{R}^n$ correspond to a value of the state at the *i*-the Collocation Point.

4.2 Pseudospectral Discretizations

To solve the Optimal Control Problem (2.4)-(2.10), the state and control variables are approximated by expansions,

$$\boldsymbol{x}_k(\tau) \approx \sum_{j=0}^{N} x_{jk}\ell_j(\tau), \quad \boldsymbol{u}_k(\tau) \approx \sum_{j=1}^{N} u_{jk}\ell_j(\tau), \ k = 1, \cdots, n,$$

such that $x_{jk} = x_k(\tau_j)$ and $u_{jk} = u_k(\tau_j)$. Note that Legendre polynomial is defined on the unit interval $[-1, 1]$ and, therefore, it is necessary to include the transformation affine

$$s = \frac{s_f - s_0}{2}\tau + \frac{s_f + s_0}{2}, \tag{4.8}$$

which converts the distance interval $s \in [s_0, s_f]$ into the fixed interval $\tau \in [-1, +1]$. Note that s is the distance along the center line of the track and s_0 and s_f are the end and start grid points of each subinterval.

Based on the choice of interpolation nodes, the LG, LGR, and LGL pseudospectral approaches are defined.

4.2.1 Collocation at Legendre-Gauss points

The state vectors connected to these points are designated as $\boldsymbol{x}_0$ and $\boldsymbol{x}_f$, respectively. The approximation of the state is done at the collocation points and only the point τ_0. Thus, the state approximation at the collocation points and only the point τ_0 becomes: According to [21] and applying the Legendre-Gauss pseudospectral method, the state is approximated globally as:

$$\dot{x}_k(\tau) \approx \sum_{j=0}^{N} D_{ij}^{LG} \dot{X}_j = \boldsymbol{D}^{LG} \boldsymbol{X}, \quad \boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}_0 \\ \boldsymbol{X}^{LG} \end{bmatrix}$$

where $\boldsymbol{D}^{LG}$ is the N by $N+1$ rectangular Legendre-Gauss differentiation matrix and $\boldsymbol{X}^{LG}$ is the state vector evaluating at the Legendre-Gauss points. To guarantee the continuity, the value of the state at $\tau_{N+1} = 1$ is then obtained from a Gauss quadrature:

$$\begin{aligned} \boldsymbol{x}_f = \boldsymbol{x}_0 + \int_{-1}^{1} \dot{x}_k(\tau) d\tau \quad &= \boldsymbol{x}_0 + \tfrac{s_f - s_0}{2} \sum_{j=0}^{N} w_j \dot{x}_k(\tau_i) d\tau \\ &= \boldsymbol{x}_0 + \tfrac{s_f - s_0}{2} \sum_{j=0}^{N} w_j (\boldsymbol{D}^{LG} \boldsymbol{X})_{jk} = x_0 + \frac{s_f - s_0}{2} \boldsymbol{w}_{LG} \boldsymbol{f}(\boldsymbol{X}^{LG}, \boldsymbol{U}^{LG}) \end{aligned}$$

where $\boldsymbol{w}_{LG} = \left[w_1^{LG}, w_2^{LG}, \cdots, w_N^{LG}\right]^T$ is a row vector of Gauss weights and $\boldsymbol{s}_{LG} = [s_1, s_2, \cdots, s_N]^T$ to just be the distance points in the $(-1, 1)$ domain connected to the Legendre-Gauss points. Furthermore, the cost is approximated using a Gauss quadrature,

$$J \approx \Phi(\boldsymbol{x}_f, s_f) + \frac{s_f - s_0}{2} \sum_{i=1}^{N} w_i^{LG} L(\boldsymbol{X}_i^{LG}, \boldsymbol{U}_i^{LG}, s_i).$$

The problem (2.4)-(2.10) can be rewritten as follows:

$$\boldsymbol{X}^{LG} \in \mathbb{R}^{N \times n}, \boldsymbol{x}_0 \in \mathbb{R}^n, \boldsymbol{x}_f \in \mathbb{R}^n, \boldsymbol{U} = \boldsymbol{U}^{LG} \in \mathbb{R}^{N \times n}$$

$$\begin{cases} \min \quad J(\boldsymbol{u}) = \Phi(\boldsymbol{x}_f, s_f) + \frac{s_f - s_0}{2} \sum_{i=1}^{N} w_i^{LG} L(\boldsymbol{X}_i^{LG}, \boldsymbol{U}_i^{LG}, s_i), \\ \text{s. t.} \quad \boldsymbol{D}^{LG} \begin{bmatrix} \boldsymbol{x}_0 \\ \boldsymbol{x} \end{bmatrix} = \frac{s_f - s_0}{2} \boldsymbol{f}(\boldsymbol{X}^{LG}, \boldsymbol{U}^{LG}, s^{LG}), \\ \quad \boldsymbol{x}_f = \boldsymbol{x}_0 + \frac{s_f - s_0}{2} \boldsymbol{w}_{LG}^T f(\boldsymbol{X}^{LG}, \boldsymbol{U}^{LG}, \mathbf{s}^{LG}), \\ \quad \Psi(s_f, \boldsymbol{x}_f) = 0, \\ \quad \boldsymbol{g}(\boldsymbol{X}^{LG}, \boldsymbol{U}^{LG}, \mathbf{s}^{LG}) \leq 0. \end{cases} \tag{4.9}$$

4.2.2 Collocation at Legendre-Gauss-Lobatto points

Denoting the LGL points by $(\tau_1, ..., \tau_N)$ where $\tau_1 = 1$ e $\tau_N = 1$ the state is approximated as:

$$x_k(\tau) \approx \sum_{j=1}^{N} x_{jk} \ell_j(\tau), \quad k = 1, \cdots, n, \tag{4.10}$$

Thus, the state derivative reads:

$$\dot{x}_k(\tau) \approx \sum_{j=1}^{N} D_{ij}^{LGL} \dot{X}_j = \boldsymbol{D}^{LGL} \boldsymbol{X}^{LGL},$$

and, from (4.8), the problem (2.4)-(2.10) becomes:

$$\boldsymbol{X} = \boldsymbol{X}^{LGL} \in \mathbb{R}^{N \times n}, \boldsymbol{U} = \boldsymbol{U}^{LGL} \in \mathbb{R}^{N \times n}$$

$$\begin{cases} \min \quad J(\boldsymbol{u}) = \Phi(\boldsymbol{x}_f, s_f) + \frac{s_f - s_0}{2} \sum_{i=1}^{N} w_i^{LGL} L(\boldsymbol{X}_i^{LGL}, \boldsymbol{U}_i^{LGL}, s_i), \\ \text{s. t.} \quad \boldsymbol{D}^{LGL} \boldsymbol{X}^{LGL} = \frac{s_f - s_0}{2} \boldsymbol{f}(\boldsymbol{X}^{LGL}, \boldsymbol{U}^{LGL}, s^{LGR}), \\ \quad \Psi(s_f, \boldsymbol{x}_f) = 0, \\ \quad \boldsymbol{g}(\boldsymbol{X}^{LGL}, \boldsymbol{U}^{LGL}, \mathbf{s}^{LGL}) \leq 0. \end{cases} \tag{4.11}$$

4.2.3 Collocation at Legendre-Gauss-Radau points

By including a non-collocated point $\tau_{N+1} = 1$, which is utilized to estimate the state only at the final time, the state derivative is approximated as follows:

$$\dot{x}_k(\tau) \approx \sum_{j=1}^{N+1} D_{ij}^{LGR} \dot{X}_j = \boldsymbol{D}^{LGR} \boldsymbol{X}^{LGR} + \boldsymbol{D}^{LGR} \boldsymbol{x}_f,$$

where $\boldsymbol{D}^{LGR}$ is the rectangular Legendre-Gauss-Radau differentiation matrix with dimensions N by $N+1$, the row vector $\boldsymbol{x}_f \in \mathbb{R}^n$ represents the non-collocated discretization point. Using equation 4.3 and the transformation (4.8), the problems (2.4)-(2.10) is given by:

$$\boldsymbol{X}^{LGR} \in \mathbb{R}^{N\times n}, \boldsymbol{x}_f \in \mathbb{R}^n, \boldsymbol{U} = \boldsymbol{U}^{LGR} \in \mathbb{R}^{N\times n}$$

$$\begin{cases} \text{min} \quad J(\boldsymbol{u}) = \Phi(\boldsymbol{x}_f, s_f) + \frac{s_f - s_0}{2} \sum_{i=1}^{N} w_i^{LGR} F(\boldsymbol{X}_i, \boldsymbol{U}_i, s_i), \\ \text{s. t.} \quad \boldsymbol{D}^{LGR} \begin{bmatrix} \boldsymbol{X}^{LGR} \\ \boldsymbol{x}_f \end{bmatrix} = \frac{s_f - s_0}{2} \boldsymbol{f}(\boldsymbol{X}^{LGR}, \boldsymbol{U}^{LGR}, s^{LGR}), \\ \quad\quad \Psi(s_f, \boldsymbol{x}_f) = 0, \\ \quad\quad \boldsymbol{g}(\boldsymbol{X}^{LGR}, \boldsymbol{U}^{LGR}, \mathbf{s}^{LGR}) \leq 0. \end{cases} \tag{4.12}$$

Note that the matrix multiplication $\boldsymbol{D}^{LGR} \begin{bmatrix} \boldsymbol{x}^{LGR} \\ \boldsymbol{x}_f \end{bmatrix}$ yields a N by N resulting matrix, therefore the equivalence holds for both the left- and right-hand side dimensions.

Chapter 5

Lap Time Simulation

In this chapter, the Pseudospectral methods are reviewed and respective results compared in terms of Optimal Lap Time and optimal trajectory of the tracks.

5.1 Software implementation

The code, in Python, used in this work was adapted from the open source repository [12]. The original tracks are all Formula 1 tracks, from where the centre line coordinates and the widths of the track were extracted. To create a larger dataset, augmentation techniques, namely rotation, scaling and shear matrix, were used on the original tracks. The centre line coordinates and the widths of the track were used as input for the Optimal Control Problem. To solve the Optimal Control Problem, after converting to an Nonlinear Programming Problem using the pseudospectral methods, the open-source framework called CasADi [1] was used. CasADi implements automatic differentiation in forward and backward modes on sparse matrix-valued computational graphs, which allows efficient computation of gradients, Jacobians, and Hessians of complex functions. This framework also provides a variety of Nonlinear Programming Problem solvers, including IPOPT and SNOPT. In this case, IPOPT method was used. IPOPT is an Interior Point OPTimizer that is known for its scalability, which is very important to handle large complex Nonlinear Programming Problems, and it is designed to find locally optimal solutions, as it employs advanced algorithms to ensure convergence to a solution. IPOPT is used together with CasADi for solving Nonlinear Programming Problems, as CasADi provides the symbolic representation, while IPOPT performs the numerical optimization to find the optimal solution for the Nonlinear Programming Problem given. The parameters used as the IPOPT solver options were the maximum number of iterations as 2000 and the error tolerance of 1e-7.

The main steps for solving the Optimal Control Problem are as follows:

- Start by defining the symbolic variables;
- Define dynamic system equations and the cost function;
- Discretization of the time;
- Discretization of the problem;
- Build the Nonlinear Programming Problem;
- Define solver options and create the IPOPT solver instance;
- Define the initial guess and bounds;
- Solving the Nonlinear Programming Problem;
- Extraction of the solution.

5.2 Methods Review

The reasons for using Legendre-Gauss methods are that this method provides high accuracy on the results, the mathematical formulation of the method is relatively simple, it is well-suited for integrating smooth functions over a finite interval and the nodes are distributed so that they tend to concentrate more points where the integrand is rapidly changing.

In the Legendre-Gauss-Radau method, the inclusion of one endpoint of the domain [-1,1] makes it well-suited for problems with boundary conditions at one end of the interval, it is especially useful for solving boundary value problems in mathematics and physics, reduced dimensionality (since it has one endpoint, often requires fewer integration points to achieve the same level of accuracy as Legendre-Gauss method) and provides accurate approximations for integrals of polynomials until a certain degree.

For the Legendre-Gauss-Lobatto method, the inclusion of both endpoints makes it particularly beneficial when dealing with boundary conditions at both ends of the interval. This also provided higher accuracy near both boundary points, which is very helpful in problems that have certain boundary conditions or mixed conditions. As in Legendre-Gauss-Radau method, the quadrature can provide accurate approximations for integrals of polynomials up to a certain degree.

The computational time of each method depends on the objective of the problem that is being handled:

- Legendre-Guass-Radau method can have some computational advantages due to the fact that it has one of the endpoints and in problems where there are conditions specific at the endpoints, the Legendre-Gauss-Radau method may require fewer points than Legendre-Gauss or Legendre-Gauss-Lobatto methods near the boundary, resulting in shorter computation times in those regions.

- Legendre-Gauss is efficient when used with a small number of interior points, especially in smooth functions, and it may require fewer quadrature points than the other methods.

- Legendre-Gauss-Lobatto, in order to get the same level of accuracy as the other methods in the interior, may require more quadrature points. However, in problems with boundary conditions or singularities at both endpoints, this method may be more efficient.

To conclude, all these methods have their advantages in niche conditions and ultimately the choice of the method to use is a trade-off between accuracy and computational efficiency.

So, if the problem has:

- boundary conditions at both endpoints, the best method is Legendre-Gauss-Lobatto

- boundary conditions at one endpoints, the best method is Legendre-Gauss-Radau

- no significant concern for boundary conditions. the best method is Legendre-Gauss

5.3 Optimal Control Problem solutions comparison

The Lap-Time Optimal Control Problem (2.4)-(2.10) was solved numerically using the LG and LGR pseudospectral formulations, described by (4.9) and (4.12), respectively. The parameter values given in Table 2 were uses to solve the problem.

Symbol	Description	Value
ρ	Air density	1.2041 kg/m^3
g	Gravitational acceleration	9.81 m/s^2
m	Mass of the vehicle	704 kg
l	Length of the vehicle	5.065 m
β_{max}	Maximum side slip angle	3.14 rad
β_{min}	Minimum side slip angle	-3.14 rad
l_f	Distance between center of mass and front axle	1.5 m
l_r	Distance between center of mass and rear axle	1.4 m
l_{w_f}	Length of the front axle	1.6 m
l_{w_r}	Length of the rear axle	1.6 m
A	Frontal area of the vehicle	1.805 m
h_{cog}	Height of the center of gravity	0.298 m
J_{zz}	Mass moment of inertia	1200 kgm^2
c_d	Drag coefficient	0.75
f_r	Rolling resistance coefficient	0.013
$k_{drive,front}$	Driving force at the front axle	0.0
$k_{break,front}$	Braking force at the front axle	0.6
k_{roll}	Roll moment at the front axle	0.5
P_{max}	Maximal engine power	230 kW
$F_{drive,max}$	Maximal drive force	7000 N
t_{drive}	Time constant for acceleration dynamic	0.05s
t_{brake}	Time constant for braking dynamic	0.05s
t_{delta}	Time constant for steering dynamic	0.2s
$F_{drive,max}$	Maximal brake force	20000 N
δ_{max}	maximal steering angle	0.193 rad
v_{max}	Maximum velocity	91.66667 m/s
k_{max}	Maximum curvature of the vehicle	0.12 rad/m

Table 2: Values used as the parameters of the vehicle.

Now, to better acknowledge the differences in terms of computation time and trajectory variations, apart from the differences that were explained previously, real examples to compare the methods will be introduced by using graphics. To do so, 25 Formula One tracks will be used with Legendre-Gauss and Legendre-Gauss-Radau methods so that a fair comparison is made and to have a comparison of the computation time of each method, the overall error that each method had in the calculation of the optimal control problem and the number of iterations required for the Optimal Control Problem to get the optimal trajectory and controls.

To see the computation time, the number of iterations of the Nonlinear Programming Problem needed to get the optimal trajectory and the error of each method (Overall Nonlinear Programming Problem error), two tables will be introduced: one for the Legendre-Gauss method and one for the Legendre-Gauss-Radau

method.

Being introduced the values of each parameter, the results obtained by running the Optimal Control Problem with CasADi's IPOPT solver for Legendre-Gauss and Legendre-Gauss-Radau methods will be introduced on Tables 3 and 4. Note that the results are specific for this vehicle's characteristics, but both methods can obtain optimal solutions for different vehicle's parameters. Also note that the tracks presented on the following tables are real-life tracks and this tracks were used to augment the dataset from 25 tracks to 3000 using geometric transformations, such as shear matrix and scaling.

Track	Computation Time	Iterations	Error
Austin	255.26 s	536	5.7605e-04
Brands Hatch	145.83 s	434	1.7214e-05
Budapest	147.45 s	393	9.0029e-04
Catalunya	216.03 s	553	9.0917e-09
Hockenheim	219.27 s	536	3.08115e-04
IMS	140.52 s	415	1.1072e-04
Melbourne	339.29 s	752	5.9491e-08
Mexico City	225.11 s	602	1.0566e-04
Montreal	220.11 s	588	9.9423e-06
Monza	231.51 s	450	1.4010e-04
Moscow Raceway	150.08 s	417	9.0961e-06
Norisring	130.03 s	625	9.0913e-09
Nuerburgring	167.09 s	377	9.5501e-04
Oschersleben	279.40 s	808	2.5079e-04
Sakhir	219.51 s	471	7.6830e-05
Sao Paulo	178.96 s	475	9.0967e-06
Sepang	217.88 s	459	4.4610e-05
Shanghai	227.90 s	490	1.5849e-05
Silverstone	314.77 s	613	6.0377e-04
Sochi	341.02 s	683	7.1023e-05
Spa Francorchamps	392.75 s	667	3.8362e-04
Spielberg	206.30 s	538	7.5453e-05
Suzuka	321.92 s	659	9.8507e-04
Yas Marina	275.02 s	588	1.4384e-08
Zandvoort	115.78 s	305	1.0178e-05

Table 3: Results obtained using Legendre-Gauss method when running the Optimal Control Problem.

Track	Computation Time	Iterations	Error
Austin	245.79 s	526	2.5052e-04
Brands Hatch	186.80 s	551	9.0916e-06
Budapest	165.11 s	438	5.1497e-04
Catalunya	253.53 s	653	9.2090e-09
Hockenheim	169.08 s	443	3.4580e-04
IMS	129.71 s	392	4.2449e-04
Melbourne	234.07 s	512	7.2925e-04
Mexico City	164.34 s	449	9.0944e-06
Montreal	145.45 s	391	3.0132e-04
Monza	403.62 s	815	9.0909e-09
Moscow Raceway	123.35 s	354	2.0623e-04
Norisring	85.99 s	432	5.7092e-04
Nuerburgring	236.47 s	539	6.3283e-05
Oschersleben	163.55 s	490	9.0913e-06
Sakhir	392.14 s	855	2.1686e-08
Sao Paulo	139.28 s	367	9.0231e-04
Sepang	246.43 s	547	2.4631e-05
Shanghai	312.92 s	683	9.0909e-06
Silverstone	267.71 s	549	1.5125e-04
Sochi	243.41 s	518	9.0911e-06
Spa Francorchamps	382.14 s	680	5.6596e-08
Spielberg	168.17 s s	468	9.0910e-06
Suzuka	258.58 s	542	9.0909e-06
Yas Marina	333.95 s	744	6.8682e-08
Zandvoort	121.52 s	347	9.0909e-06

Table 4: Results obtained using Legendre-Gauss-Radau method when running the Optimal Control Problem.

5.3.1 Results overview

Analyzing the data collected running the Optimal Control Problem, it is possible to see that the computation time, number of iterations and overall error in some tracks is better in the Legendre-Gauss method and in others the Legendre-Gauss-Radau method seems to be the better method to use.

It is possible to see that in tracks, such as Shanghai, Yas Marina and Zandvoort, that have low speed corners, the time and iterations that it takes for the Optimal Control Problem to finish is lower in the Legendre-Gauss method than in the Legendre-Gauss-Radau method. However, the error is only lower, for the Legendre-Gauss method, in the Yas Marina track.

On other tracks, such as Montreal, Mexico City and Spielberg, which are fast tracks, the time and

iterations of the Optimal Control Problem is lower in the Legendre-Gauss-Radau method but the error is lower, in the Legendre-Gauss method, for the Montreal track.

Since every track has its unique characteristics, it was expected since the beginning that this could occur. A look at some examples that will be shown next using graphical information will allow to visualize if the trajectories change significantly between methods.

5.4 Optimal Trajectories

By seeing the data shown, it is possible to make a comparison in terms of efficiency of both methods. But what about the trajectories? Are they alike? Or do the trajectories differ between the methods? That is what will be found in this section.

5.4.1 Comparison graphics

Here graphics from all tested tracks will be shown to compare the trajectories obtained by the Optimal Control Problem and see which method is better overall because the fact that one method is better in terms of overall error, time spent or number of iterations, does not mean that the resulting trajectory is 100% better, or better at all. It all comes down to what was previously discussed in the comparison of all the methods about the convergence in the middle of the domain used and the convergence taking in account one of the endpoints of the domain or taking in account both of the endpoints, which is specific of each method.

In each of the following graphics (9 to 33), note that the parts of the track that are shown are the parts where it is trickier to predict the trajectory, since these are the tightest corners, sequences of corners, or two corners from two different parts of the track. Also note that the reason only parts of the track are presented is because if the whole track was shown, it would be impossible to compare both trajectories because the image of the track would be too compressed to be able to see the differences, apart from the fact that the trajectories tend to overlap each other, so it is better to show images side to side of the same parts of the track to get a better visual comparison.

In the end, just for a final comparison, are presented a set of graphics with both trajectories to get a different comparison instead of two separate graphics.

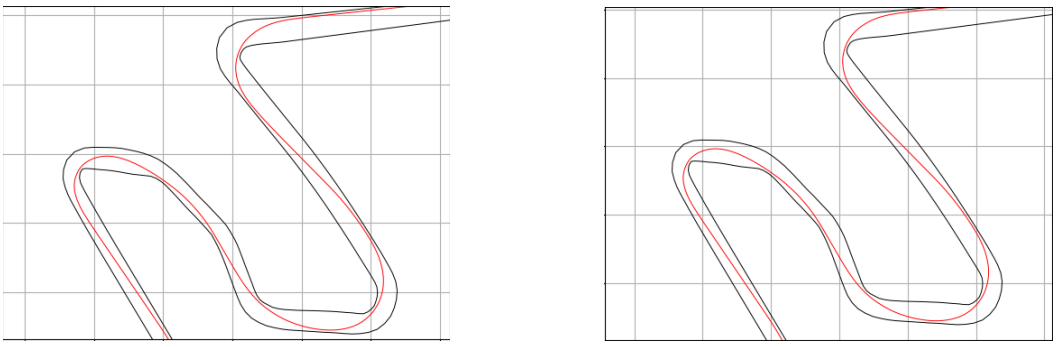

Figure 9: Optimal Trajectory of Austin track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

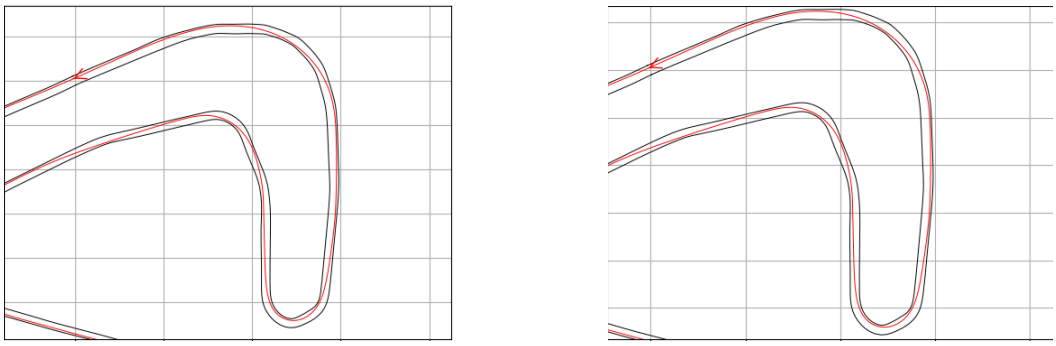

Figure 10: Optimal Trajectory of Brandshatch track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

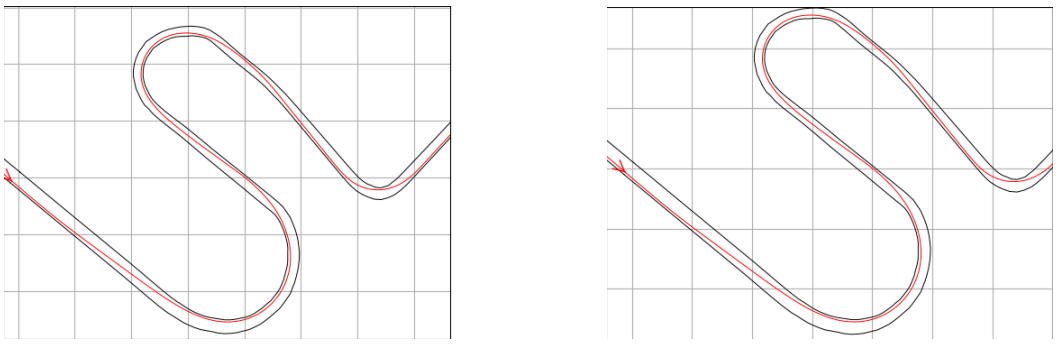

Figure 11: Optimal Trajectory of Budapest track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

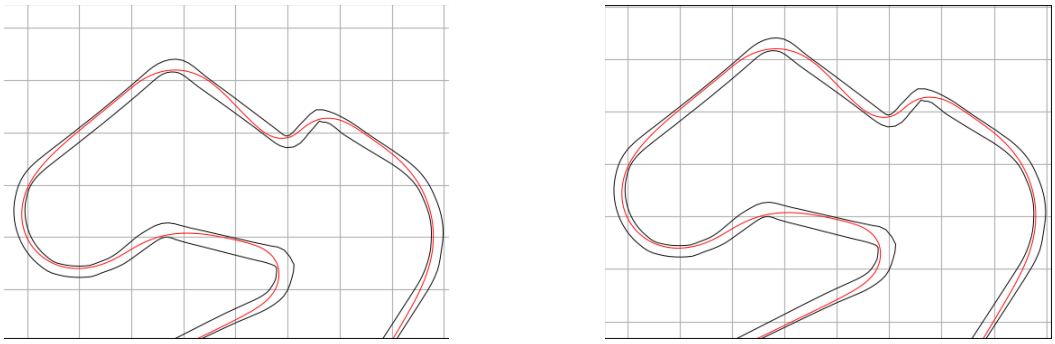

Figure 12: Optimal Trajectory of Catalunya track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

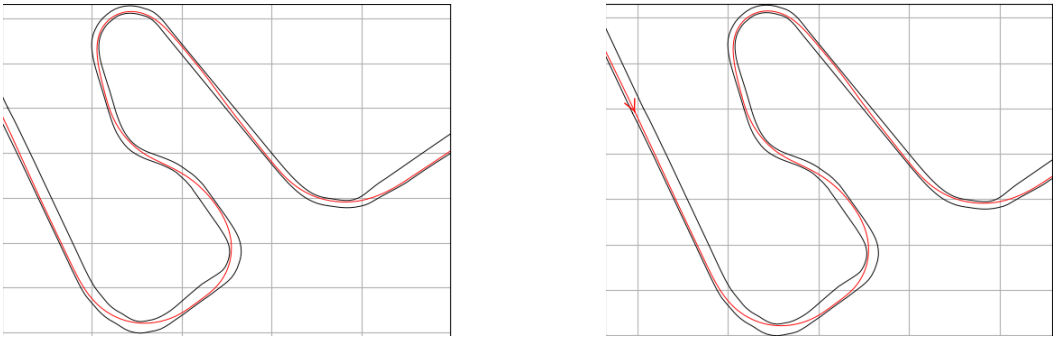

Figure 13: Optimal Trajectory of Hockenheim track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

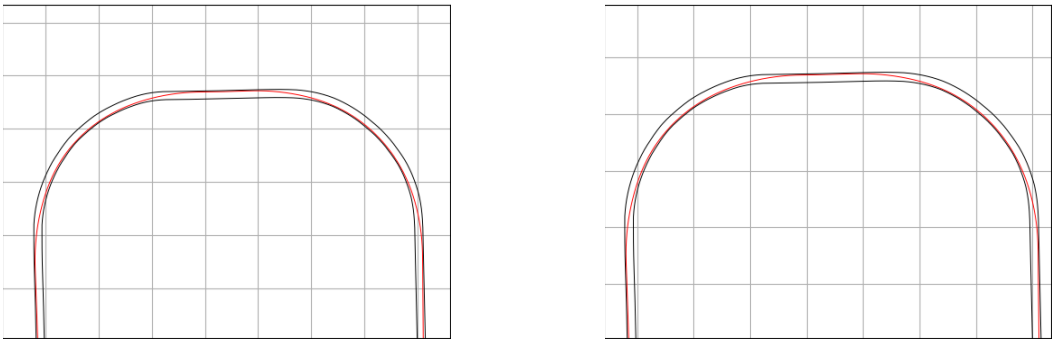

Figure 14: Optimal Trajectory of IMS track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

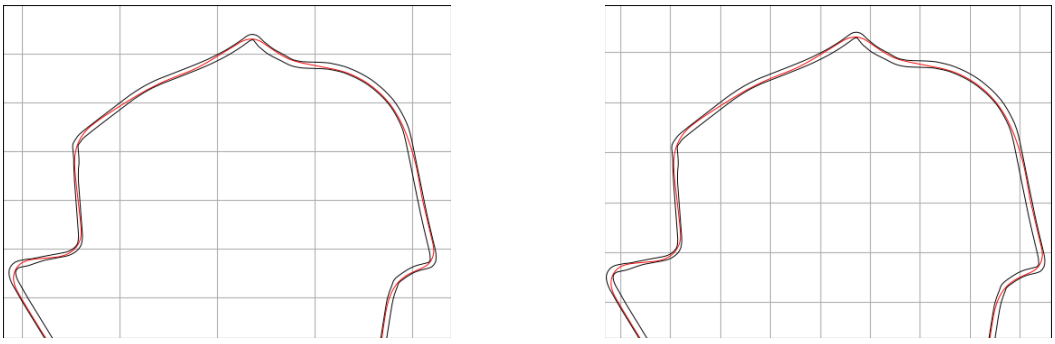

Figure 15: Optimal Trajectory of Melbourne track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

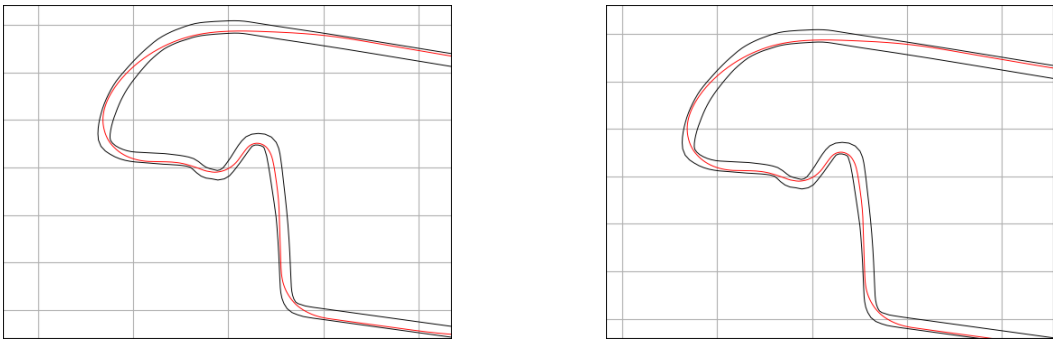

Figure 16: Optimal Trajectory of Mexico City track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

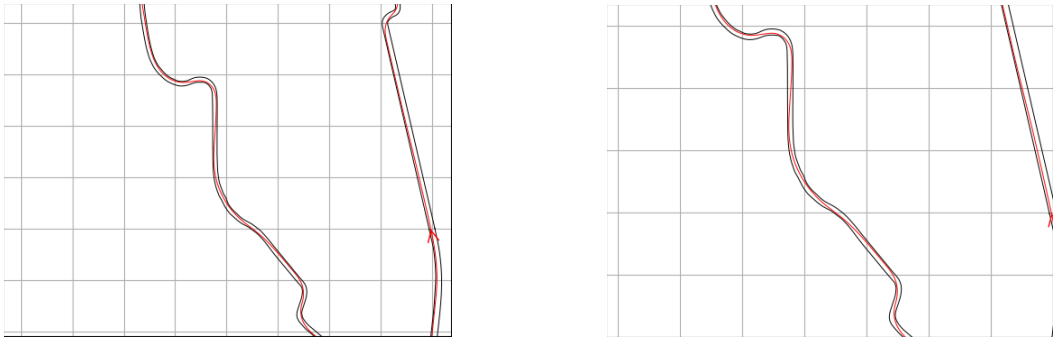

Figure 17: Montreal track with Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

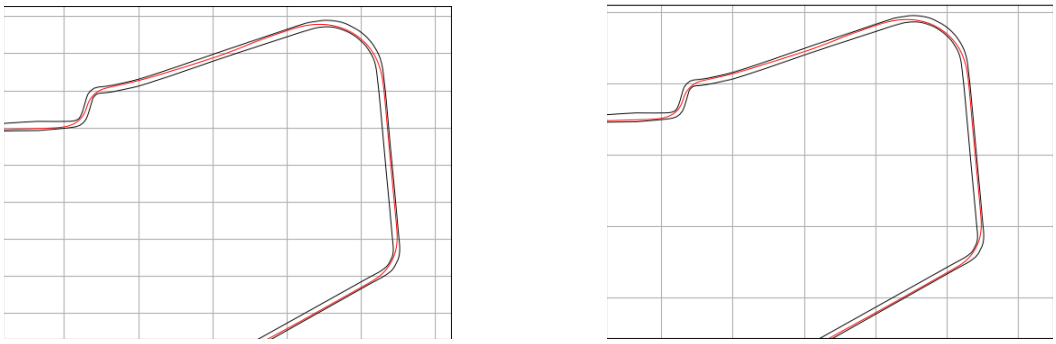

Figure 18: Optimal Trajectory of Monza track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

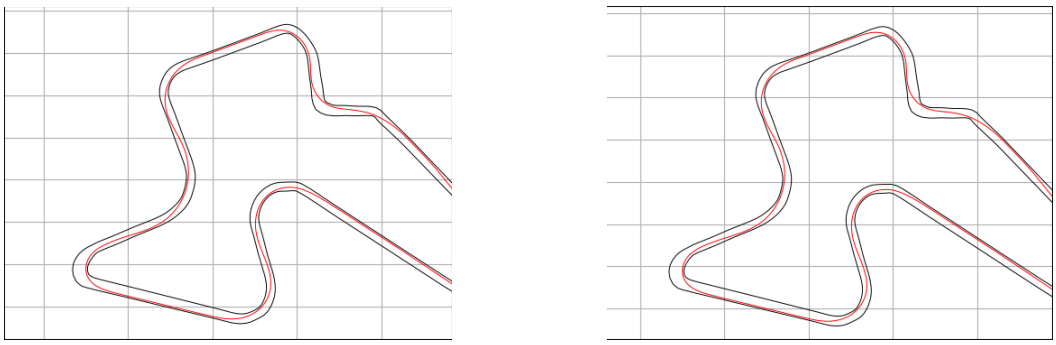

Figure 19: Optimal Trajectory of Moscow Raceway track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

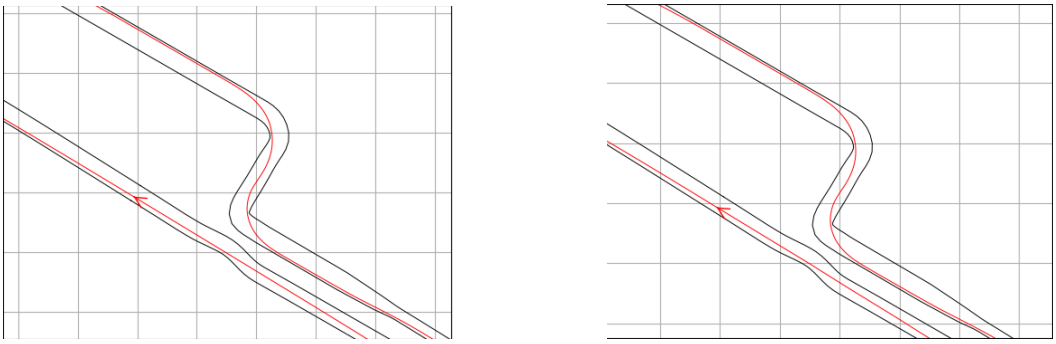

Figure 20: Optimal Trajectory of Norisring track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

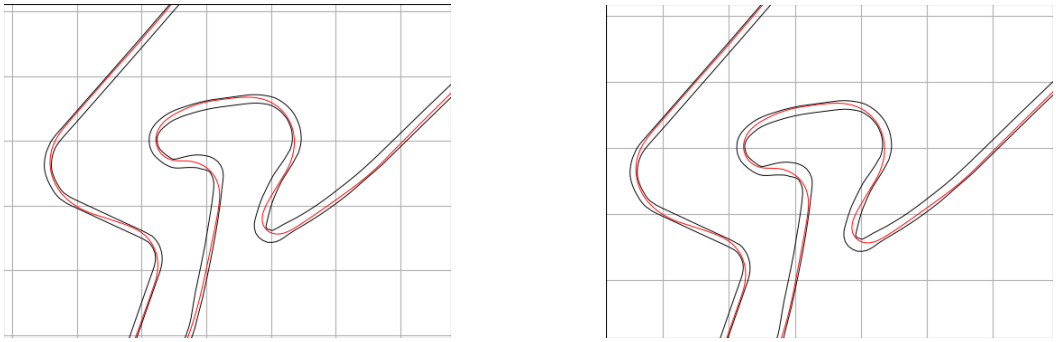

Figure 21: Optimal Trajectory of Nuerburgring track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

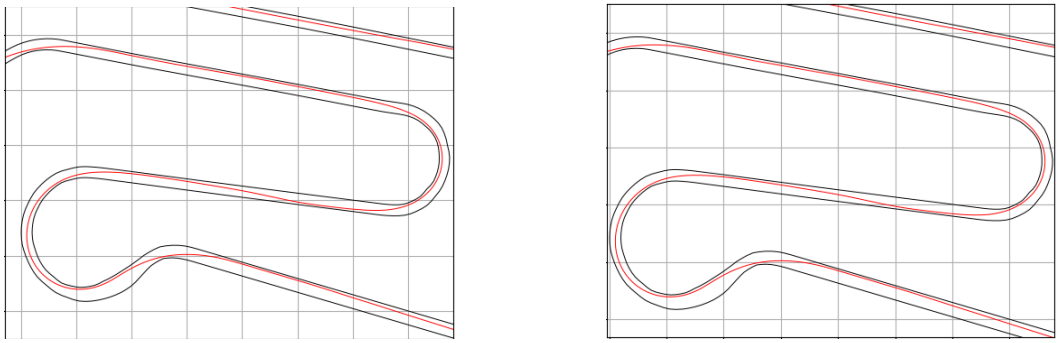

Figure 22: Optimal Trajectory of Oschersleben track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

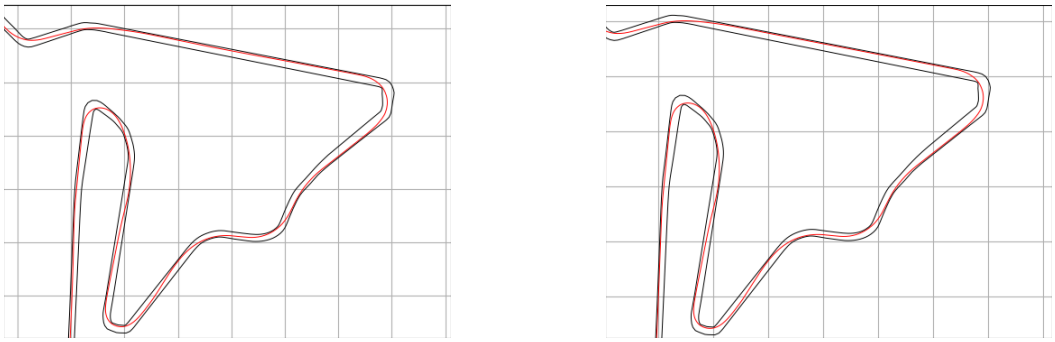

Figure 23: Optimal Trajectory of Sakhir track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

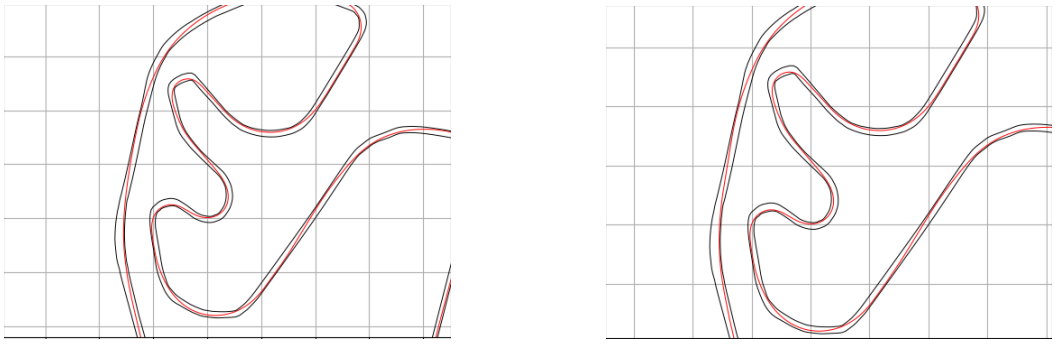

Figure 24: Optimal Trajectory of Sao Paulo track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

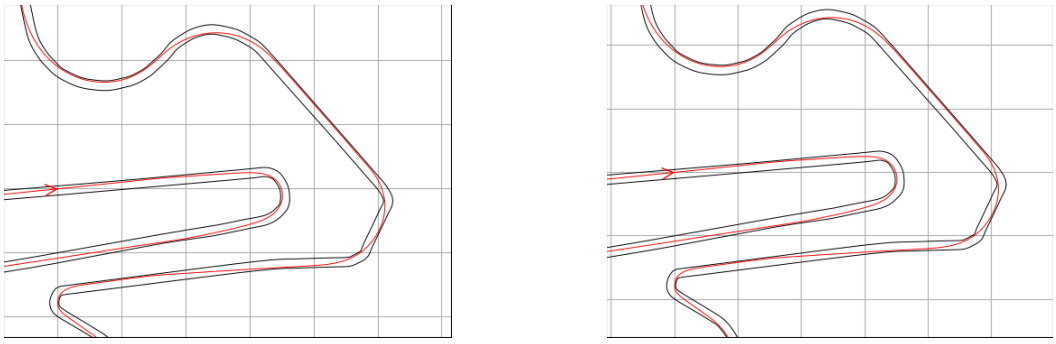

Figure 25: Optimal Trajectory of Sepang track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

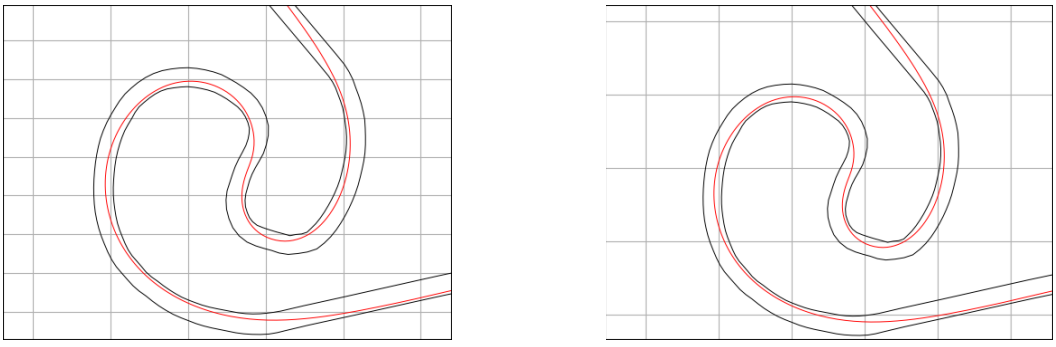

Figure 26: Optimal Trajectory of Shanghai track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

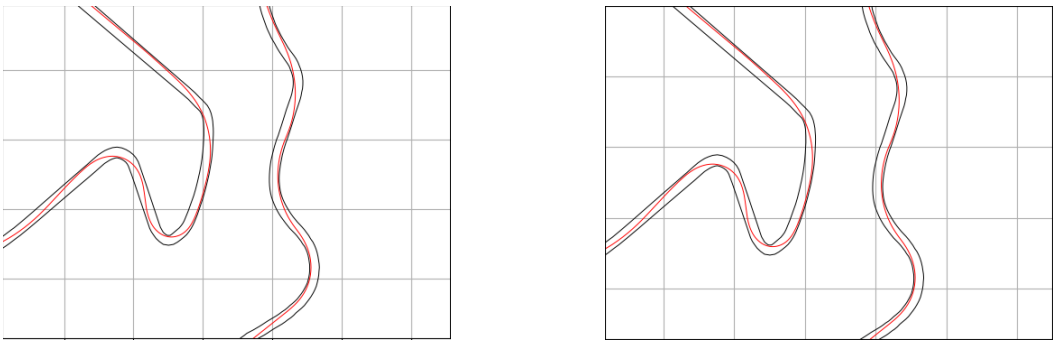

Figure 27: Optimal Trajectory of Silverstone track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

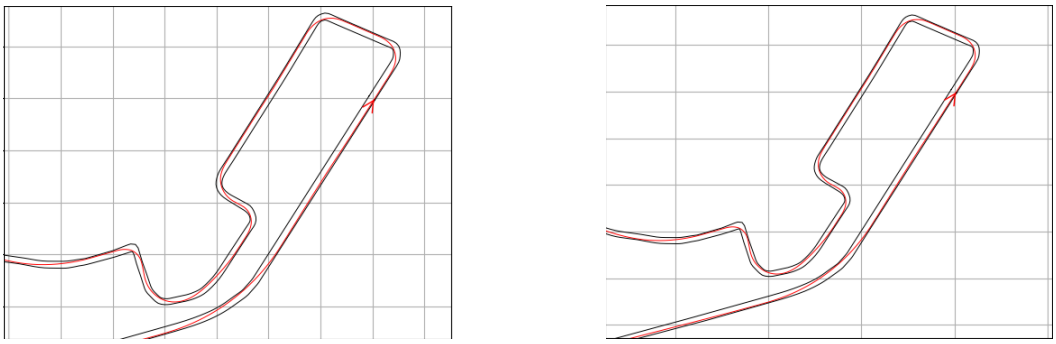

Figure 28: Optimal Trajectory of Sochi track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

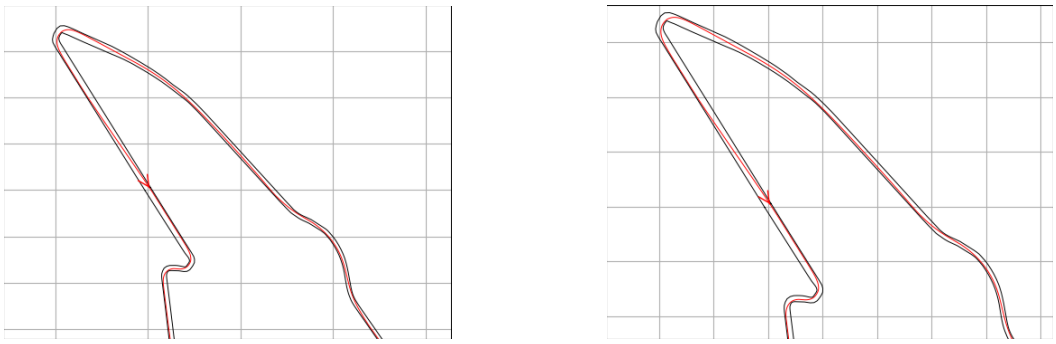

Figure 29: Optimal Trajectory of Spa Francorchamps track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

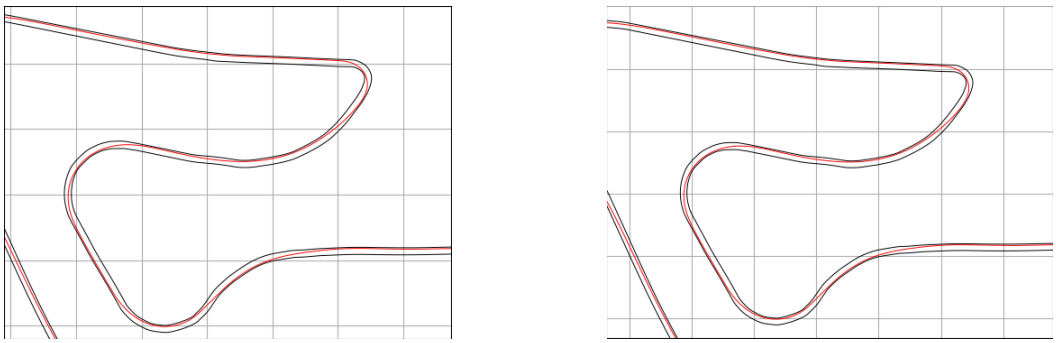

Figure 30: Optimal Trajectory of Spielberg track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

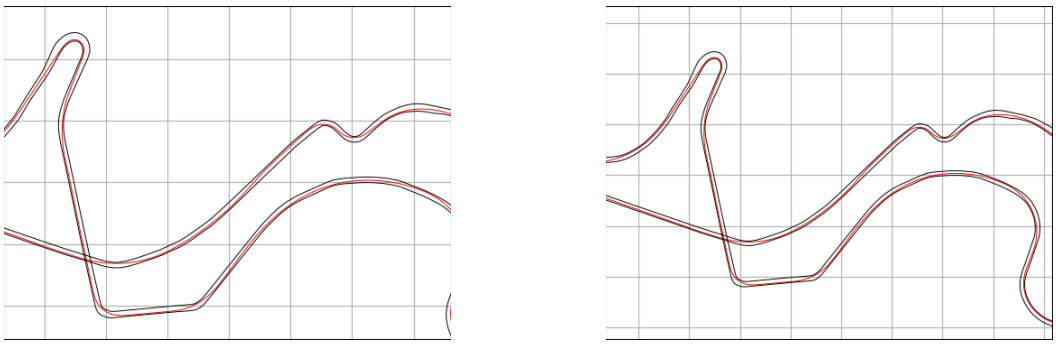

Figure 31: Optimal Trajectory of Suzuka track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

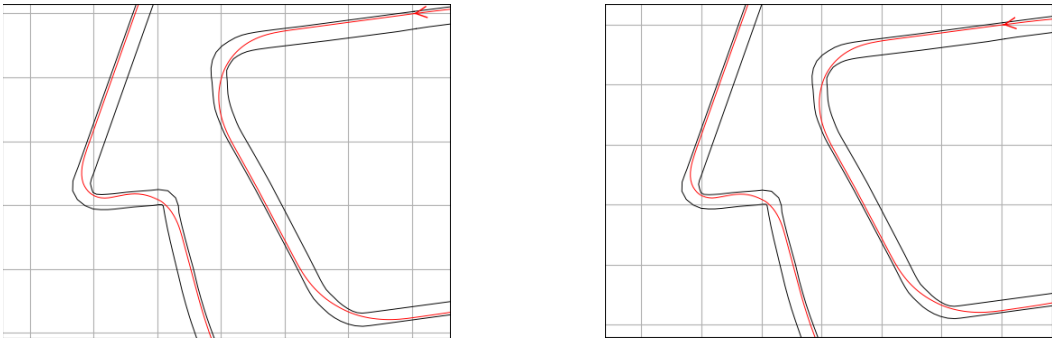

Figure 32: Optimal Trajectory of Yas Marina track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

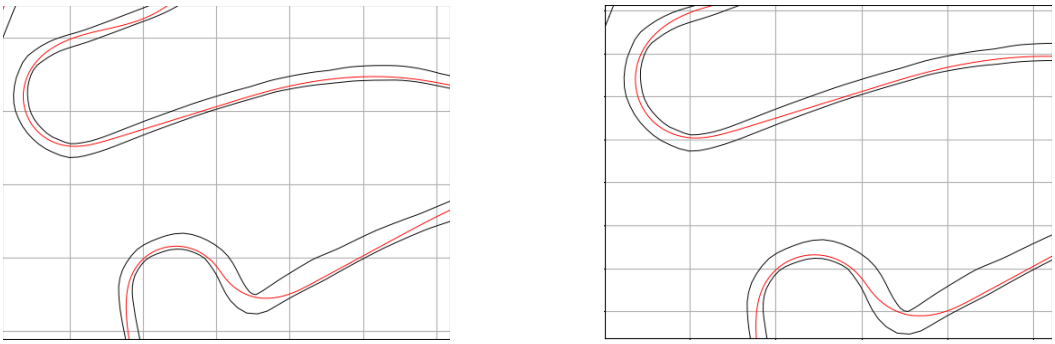

Figure 33: Optimal Trajectory of Zandvoort track using Legendre-Gauss method (left) and Legendre-Gauss-Radau method (right)

From all the graphics until now of all the tracks tested (from Figure 9 to Figure 33), it looks like every single one of the tracks have similar trajectories, or the trajectories are so close from each other that it is impossible to see any difference between them.

Now a closer look will be taken to see the differences between the trajectories of both methods.

Note that the following graphics (from Figure 35 to Figure 45) where both trajectories will be visible, the images are from the full tracks zoomed in.

Also, note that the continuous trajectory (red trajectory) represents the minimum lap time trajectory of the Legendre-Gauss method band the dashed trajectory (green trajectory) refers to the minimum lap time trajectory obtained using the Legendre-Gauss-Radau method.

Figure 34: Suzuka Full Track with both trajectories

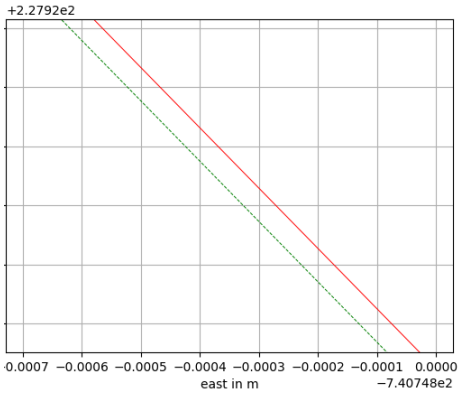

Figure 35: Suzuka Straight

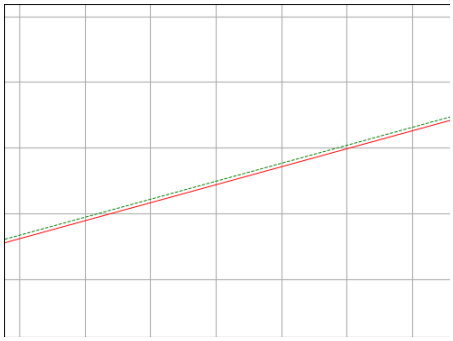

Figure 36: Suzuka Corner

As it can be seen by Figure 35 of this track, in the straights the trajectories differ significantly, by using enough zoom in, as it is possible to see by the scale of the Figure 35. On corners, for example the one on Figure 36, the trajectories differ slightly, as it can be seen by zooming on the middle of the corner.

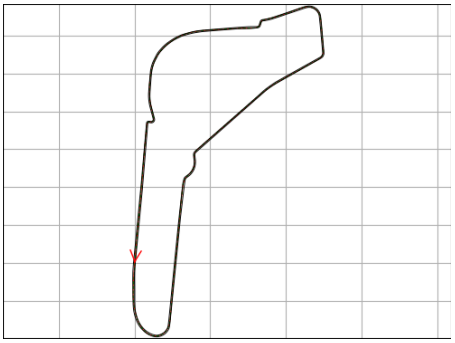

Figure 37: Monza Full Track

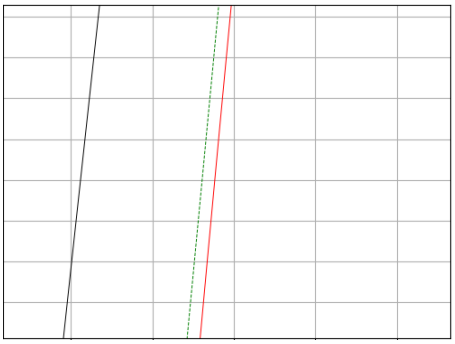

Figure 38: Monza Straight Line

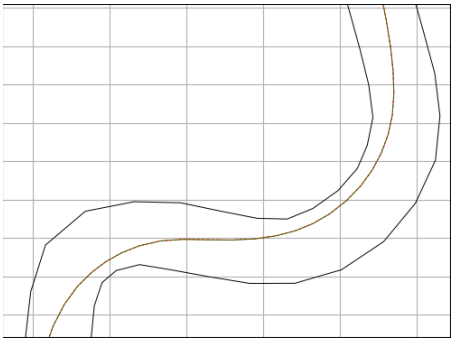

Figure 39: Monza Corner

Figure 40: Detail of Monza Corner

From the graphics of the Monza track, it is possible to see in many straights what happens in Figure 38, where the trajectories differ slightly from each other and where the red trajectory is the Legendre-Gauss method resulting trajectory and the green trajectory is the Legendre-Gauss-Radau method trajectory. In the corners, such as the first corner of the track (represented in Figure 39), the trajectories overlap each other at first sight. However, if zoomed in close enough, it is possible to see that the trajectories do differ along the corner, as it is possible to see on Figure 40.

The following graphics are a few more examples to show the differences between the trajectories which raised curiosity because of the discrepancies of computation time and iterations.

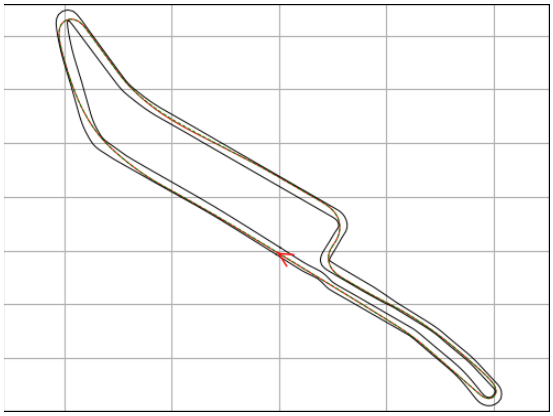

Figure 41: Norisring Full Track with both trajectories

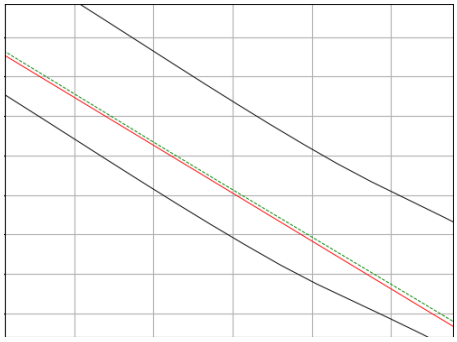

Figure 42: Norisring Straight

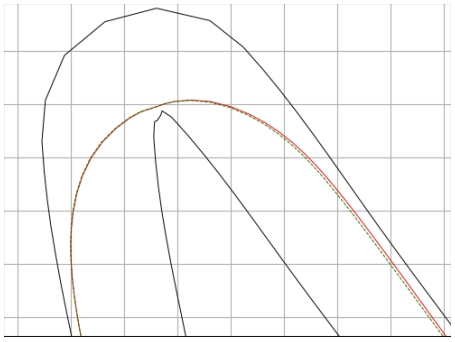

Figure 43: Norisring Corner

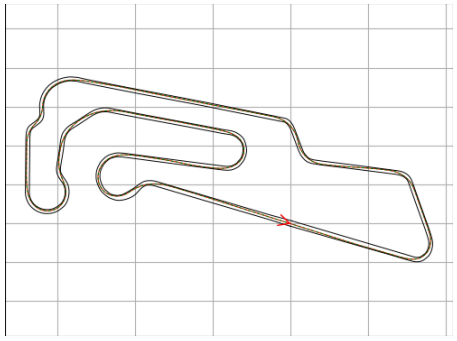

Figure 44: Oschersleben Full Track

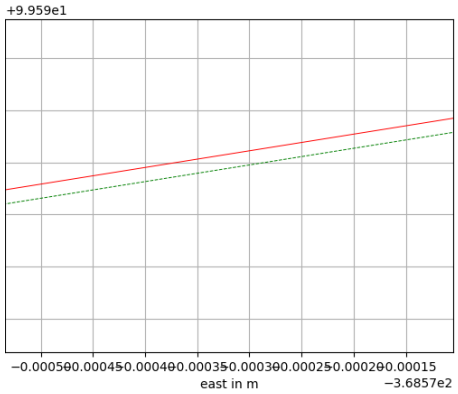

Figure 45: Oschersleben corner exit zoomed in

The reason for showing most of these examples of full tracks, corners and straights, is because these are the ones where the number of iterations and time spent on the Optimal Control Problem differ the most. As said, the differences in terms of the computation time and iterations of the Optimal Control Problem raised some curiosities to why the differences were so significant and if that would imply that the trajectories would differ that much or if the faster method was just more efficient overall for the characteristics of this specific tracks compared to the other method.

5.5 Comparison of Resulting Optimal Parameters

For a comparison of the trajectory and control parameters obtained by running the Optimal Control Problem, Tables 5 and 6 will be introduced with the coordinates of the trajectory, velocity (v), curvature (k), yaw angle velocity (ψ) and side slip angle (β). The first table (Table 5) refers to the parameters from Legendre-Gauss method and the second table (Table 6) refers to the parameters from the Legendre-Gauss-Radau method.

The results shown in both tables are from Suzuka track using the same input parameters on both methods. As it will be possible to verify on Tables 5 and 6, there are no significant differences between the results obtained by Legendre-Gauss and Legendre-Gauss-Radau methods, as it was possible to assume by seeing the graphics presented previously, even though the Legendre-Gauss-Radau had a lower error, took less time to compute and fewer iterations for the Optimal Control Problem to get the results compared to Legendre-Gauss method. This results prove that, for the track of Suzuka, the Legendre-Gauss-Radau method is more efficient and precise in obtaining an optimal trajectory and controls for the vehicle, as it takes less time to compute and has a lower overall error.

$x(m)$	$y(m)$	$v(m/s)$	$\kappa(rad/m)$	$\psi(rad)$	$\beta(rad)$
7.0429380	3.5803563	51.0639552	-0.0021360	-2.3819097	0.0131800
9.1053304	1.3950423	51.3248844	-0.0021435	-2.3904802	0.0134200
11.1532200	-0.8030963	51.4536192	-0.0021350	-2.3947582	0.0136300
13.1866683	-3.0139227	51.7059053	-0.0020611	-2.4031694	0.0137200
15.2059620	-5.2371100	51.8305909	-0.0020043	-2.4072343	0.0135800
17.2116537	-7.4721071	52.0757087	-0.0018235	-2.4149081	0.0131100
19.2045748	-9.7181295	52.1970558	-0.0017140	-2.4184451	0.0122600
21.1858083	-11.9741841	52.4360524	-0.0014563	-2.4247908	0.0110900
23.1566172	-14.2391318	52.5544295	-0.0013221	-2.4275690	0.0097200
25.1183375	-16.5117799	52.7874665	-0.0010719	-2.4323447	0.0082900
27.0722577	-18.7909860	52.9027992	-0.0009590	-2.4343754	0.0069600
29.0195125	-21.0757469	53.1294770	-0.0007893	-2.4378533	0.0058300
30.9610173	-23.3652548	53.2415468	-0.0007230	-2.4393655	0.0049700
32.8974496	-25.6589102	53.4615641	-0.0006392	-2.4420782	0.0043900
34.8292783	-27.9562964	53.5702830	-0.0006085	-2.4433258	0.0040500
36.7568275	-30.2571230	53.7836690	-0.0005574	-2.4456593	0.0038500
38.6803620	-32.5611523	53.8891097	-0.0005294	-2.4467460	0.0037100
40.6001771	-34.8681221	54.0961418	-0.0004449	-2.4487072	0.0035300
42.5166746	-37.1776813	54.1984720	-0.0003898	-2.4495418	0.0032100
44.4304117	-39.4893475	54.3994862	-0.0002362	-2.4508222	0.0027000

Table 5: Resulting parameters from Optimal Control Problem using Legendre-Gauss method

$x(m)$	$y(m)$	$v(m/s)$	$\kappa(rad/m)$	$\psi(rad)$	$\beta(rad)$
7.0453257	3.5824428	51.0642745	-0.0021359	-2.3819081	0.0131800
9.1077235	1.3971294	51.3251921	-0.0021445	-2.3904798	0.0134200
11.1556140	-0.8010122	51.4539175	-0.0021371	-2.3947609	0.0136300
13.1890474	-3.0118552	51.7061577	-0.0020678	-2.4031888	0.0137300
15.2082890	-5.2350912	51.8308095	-0.0020142	-2.4072703	0.0136100
17.2138589	-7.4701971	52.0758151	-0.0018425	-2.4150012	0.0131700
19.2065441	-9.7164265	52.1970919	-0.0017385	-2.4185818	0.0123700
21.1873724	-11.9728335	52.4359084	-0.0014933	-2.4250503	0.0112600
23.1575442	-14.2383324	52.5541865	-0.0013655	-2.4279087	0.0099500
25.1183308	-16.5117857	52.7870166	-0.0011263	-2.4328810	0.0085900
27.0709617	-18.7921006	52.9022476	-0.0010178	-2.4350248	0.0073100
29.0165274	-21.0783113	53.1287446	-0.0008508	-2.4387457	0.0062300
30.9559230	-23.3696263	53.2407340	-0.0007833	-2.4403796	0.0054000
32.8898397	-25.6654338	53.4606352	-0.0006865	-2.4433110	0.0048100
34.8188022	-27.9652686	53.5693124	-0.0006457	-2.4446431	0.0044200
36.7432383	-30.2687513	53.7826787	-0.0005633	-2.4470655	0.0041300
38.6635636	-32.5755153	53.8881307	-0.0005170	-2.4481457	0.0038400
40.5802616	-34.8851383	54.0952414	-0.0003917	-2.4499722	0.0034500
42.4939426	-37.1970916	54.1976229	-0.0003199	-2.4506839	0.0029100
44.4053866	-39.5107038	54.3987354	-0.0000982	-2.4515279	0.0021700

Table 6: Resulting parameters from Optimal Control Problem using Legendre-Gauss-Radau method

From these tables, it is possible to see that the coordinates x and y of the trajectory slightly change, as well as the controls of velocity v, curvature k, yaw angle velocity ψ and side slip angle β, between both methods. This confirms the slight differences that were presented on Figures 35 and 36.

Chapter 6

Conclusions

In this thesis an Optimal Control Problem was formulated with constraints imposed by the track and vehicle models used, which accurately capture the dynamics of the vehicle moving around a racing track.

The Optimal Control Problem was then solved using two Pseudospectral methods, namely the Legendre-Gauss and Legendre-Gauss-Radau. Another Pseudospectral method (Legendre-Gauss-Lobato) was also introduced but not used to solve the Optimal Control Problem, as it was considered only for the theoretical comparison between the three pseudospectral methods based on Legendre polynomials that were introduced throughout the thesis.

The theoretical comparison served as a way to exploit the differences in the resolution of the Optimal Control Problem between the three methods, since each track configuration has its own advantages and disadvantages for each method. This helped in introducing the idea that the trajectories that would result from running the Optimal Control Problem would probably differ from track to track, depending on the characteristics of the tracks.

On Tables 5 and 6 it was possible to check the divergences, for example of the x and y coordinates, in terms of the Optimal Control Problem trajectories and controls resultant from each of the methods, where it is possible to see that in some tracks, such as Shanghai, Yas Marina and Zandvoort, the number of iterations and the time it takes to finish running the Optimal Control Problem is quite higher in one of the methods compared to the other, and in other tracks, such as Montreal, Mexico City and Spielberg, it is the exact opposite. This was caused by the characteristics of the tracks referred previously, which made one method be better than the other, since the constraints on the end of the intervals might have influenced in some tracks and on others the interior of the time interval might have been the important part of the interval for the collocation points. Then, on the various graphics, especially from Figure 38 to Figure 45, a closer comparison of the actual differences in terms of the trajectories was presented.

By comparing the Optimal Control Problem results between these different direct collocation methods, this thesis aimed to analyze the best numerical technique for solving optimal control problems in the context of vehicle trajectory planning. Also a large database of optimal trajectories was created to train neural networks that, with the Optimal Control Problem results given, will be able to obtain the optimal trajectory and controls for tracks that are not part of the initial database with high accuracy. By using the new method implemented, it was possible to reduce the computation time of the Optimal Control Problem for 3000 tracks in 3 hours and 30 minutes.

References

[1] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl. CasADi – A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1): 1–36, 2019. doi: 10.1007/s12532-018-0139-4.

[2] R. Bellman. Dynamic programming and a new formalism in the theory of integral equations. *Proc. Natl. Acad. Sci. USA*, 41:31–34, 1955. ISSN 0027-8424. doi: 10.1073/pnas.41.1.31.

[3] R. Bellman. Dynamic programming. *Science*, 153(3731):34–37, 1966.

[4] D. P. Bertsekas. Nonlinear programming. *Journal of the Operational Research Society*, 48(3): 334–334, 1997.

[5] J. T. Betts. *Practical methods for optimal control and estimation using nonlinear programming*. SIAM, 2010.

[6] P. Casanova. *On minimum time vehicle manoeuvring: the theoretical optimal lap*. PhD thesis, Cranfield University, 2000.

[7] A. H. Fabian Christ, Alexander Wischnewski and B. Lohmann. Time-optimal trajectory planning for a race car considering variable tyre-road friction coefficients. *Vehicle System Dynamics*, 59 (4):588–612, 2021. doi: 10.1080/00423114.2019.1704804. URL `https://doi.org/10.1080/00423114.2019.1704804`.

[8] F. Fahroo and I. M. Ross. Advances in pseudospectral methods for optimal control. In *AIAA guidance, navigation and control conference and exhibit*, page 7309, 2008.

[9] D. Garg, M. Patterson, W. W. Hager, A. V. Rao, D. A. Benson, and G. T. Huntington. A unified framework for the numerical solution of optimal control problems using pseudospectral methods. *Automatica*, 46(11):1843–1851, 2010. ISSN 0005-1098. doi: https://doi.org/10.1016/

j.automatica.2010.06.048. URL https://www.sciencedirect.com/science/article/pii/S0005109810002980.

[10] D. Garg, M. A. Patterson, C. Francolin, C. L. Darby, G. T. Huntington, W. W. Hager, and A. V. Rao. Direct trajectory optimization and costate estimation of finite-horizon and infinite-horizon optimal control problems using a radau pseudospectral method. *Computational Optimization and Applications*, 49:335–358, 2011.

[11] D. Garg, M. A. Patterson, W. W. Hager, A. V. Rao, D. A. Benson, and G. T. Huntington. An overview of three pseudospectral methods for the numerical solution of optimal control problems. *Advances in the Astronautical Sciences*, 2017. URL https://api.semanticscholar.org/CorpusID:53686777.

[12] L. Hermansdorfer, A. Heilmeier, F. Christ, and T. Herrmann. Global racetrajectory optimization. https://github.com/TUMFTM/global_racetrajectory_optimization, 2021.

[13] T. Herrmann. Optimization-based energy strategy for autonomous electric race cars. *SIAM Journal on Control and Optimization*, 15(2):256–293, 2022. doi: https://doi.org/10.1137/0315019. URL https://mediatum.ub.tum.de/doc/1656862/1656862.pdf.

[14] S. Lenhart and J. T. Workman. *Optimal control applied to biological models*. CRC press, 2007.

[15] F. LIU. *Adaptative Mesh Refinement Methods for Solving Optimal Control problems*. PhD thesis, University of Florida, 2015.

[16] M. I. Masouleh. *Optimal control and stability of four-wheeled vehicles*. PhDthesis. University of Oxford, 217.

[17] M. Massaro and D. J. N. Limebeer. Minimum-lap-time optimisation and simulation. *Vehicle System Dynamics*, 59(7):1069–1113, 2021. doi: 10.1080/00423114.2021.1910718. URL https://doi.org/10.1080/00423114.2021.1910718.

[18] D. Morante, M. Sanjurjo Rivo, and M. Soler. A survey on low-thrust trajectory optimization approaches. *Aerospace*, 8(3), 2021. ISSN 2226-4310. doi: 10.3390/aerospace8030088. URL https://www.mdpi.com/2226-4310/8/3/88.

[19] J. Park, I. G. Jang, and S.-H. Hwang. Torque distribution algorithm for an independently driven electric vehicle using a fuzzy control method: Driving stability and efficiency. *Energies*, 11(12), 2018.

ISSN 1996-1073. doi: 10.3390/en11123479. URL https://www.mdpi.com/1996-1073/11/12/3479.

[20] L. S. Pontryagin. *Mathematical theory of optimal processes*. CRC press, 1987.

[21] A. Rao. A survey of numerical methods for optimal control. *Advances in the Astronautical Sciences*, 135, 01 2010.

[22] D. Solow. Linear and nonlinear programming. *Wiley Encyclopedia of Computer Science and Engineering*, 2007.

[23] S. Sowell and E. Taheri. *The Tiger Optimization Software - A Pseudospectral Optimal Control Package*. PhD thesis, Auburn University, 12 2022. URL https://doi.org/10.13140/RG.2.2.31472.33283.

[24] M. Veneri and M. Massaro. A free-trajectory quasi-steady-state optimal-control method for minimum lap-time of race vehicles. *Vehicle System Dynamics*, 58(6):933–954, 2020. doi: 10.1080/00423114.2019.1608364. URL https://doi.org/10.1080/00423114.2019.1608364.

[25] S. Yin, J. Wu, Y. Song Cao, S. Li, L. Petzold, and R. Serban. Optimal control by deep learning techniques and its applications on epidemic models. *J. Math. Biol.*, 86(36), 2023.

[26] H. YU. *Optimal powertrain design and control of an electric race car*. PhD thesis, Politecnico Milano, 2017.