Università degli studi di Padova
Department of Industrial Engineering

PhD course in Industrial Engineering

Curriculum in Mechanical Engineering — XXX cohort

Optimal control of road vehicles: theory and applications

Nicola Dal Bianco

Coordinator:	Ch.mo Prof. Paolo Colombo
Supervisor:	Ch.mo Prof. Roberto Lot
Co-supervisor:	Ch.mo Prof. Vittore Cossalter

October 31, 2017

Abstract

In this thesis Optimal Control (OC) of road vehicles is studied especially focusing on minimum lap time simulations. The theory underlying the most used optimal control solving techniques is described, including both the Pontryagin Maximum Principle and the reduction to Nonlinear Programming. Direct and indirect methods for optimal control problems are presented and compared against minimum lap time simulations (LTS).

Modelling of vehicles for OC-LTSs is studied in order to understand how different design choices can affect simulation outcomes. Novel multibody models of four wheeled vehicles — a GP2 car and a go-kart — for OC-LTSs are developed and validated thorough comparison with experimental data. Particular attention is dedicated to the simulation of tyre load dynamics, that is achieved by a proper modelling of the chassis and suspension motions and of the aerodynamic forces.

OC-LTSs are applied to electric vehicles too, specifically to optimise the design of an electric motorbike taking part at the Tourist Trophy Zero competition. A concise yet effective model is proposed in order to perform reliable simulations on a $60km$ long road in a reasonable amount of time. Experimental data is used to validate the model.

A direct full collocation transcription method for OCPs dealing with implicit differential equations and control derivatives is presented, moreover the structure of the resulting NLP problem is accurately described. The relationship between the first order necessary conditions and the Lagrange multipliers of the NLP and OC problems are derived under the adopted discretisation scheme. The presented transcription method is implemented into a software which is currently in use at the University of Padova to solve OC-LTSs.

Contents

Notation

Mathematical notation

Non bold letters, e.g. x, indicate scalars, bold lower case letters, e.g. $\boldsymbol{x}$, denote vectors, and upper case bold letters, e.g. $\boldsymbol{X}$, indicate matrices, except where differently specified. Force vectors, denoted by $\boldsymbol{F}$, are an exception to this rule. The i-th entry of a vector $\boldsymbol{x}$ is indicated by $\boldsymbol{x}_i$; similarly, the entry at the i-th row and j-th column of a matrix $\boldsymbol{X}$ is denoted by $\boldsymbol{X}_i^j$. Square brackets are used for vectors and matrices. When vector or matrix elements are separated by a comma, the elements are concatenated in different columns, i.e. they belong to the same row. For instance, a row vector containing elements a, b, c can be written as $[a, b, c]$. When elements are separated by a semicolon, they are concatenated within the same column, e.g. a column vector containing elements a, b, c can be written as $[a; b; c]$. The transpose of a vector $\boldsymbol{v}$ is denoted by $\boldsymbol{v}^\intercal$, e.g. $[a; b; c] = [a, b, c]^\intercal$. The binary operators $*$ and $/$ between two vectors or matrices of the same dimensions indicate respectively the element-wise multiplication and division.

Given a scalar function of a vector variable $f(\boldsymbol{x})$, $\nabla_{\boldsymbol{x}} f$ is the gradient of $f(\boldsymbol{x})$ with respect to $\boldsymbol{x}$, i.e. it is a column vector whose i-th entry is $\frac{\partial f}{\partial x_i}$. Given a vector function of a vector variable $\boldsymbol{f}(\boldsymbol{x})$, $\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}$, or equivalently $\partial_{\boldsymbol{x}} \boldsymbol{f}$, is the Jacobian matrix $\boldsymbol{J}$ of $\boldsymbol{f}$ with respect to (w.r.t) $\boldsymbol{x}$, i.e. $\boldsymbol{J}_i^j = \frac{\partial \boldsymbol{f}_i}{\partial \boldsymbol{x}_j}$. It follows from this definition that for a scalar function the gradient is the transpose of the Jacobian $\nabla_{\boldsymbol{x}} f = \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right)^\intercal$. For a scalar function of vector variables $f(\boldsymbol{x}, \boldsymbol{y})$, the Hessian matrix $\boldsymbol{H}$ of $\boldsymbol{f}$ with respect to $\boldsymbol{x}$ and $\boldsymbol{y}$ is denoted by $\frac{\partial^2 \boldsymbol{f}}{\partial \boldsymbol{x} \partial \boldsymbol{y}^\intercal} = \frac{\partial^2 \boldsymbol{f}}{\partial \boldsymbol{y}^\intercal \partial \boldsymbol{x}} = \boldsymbol{H}$, i.e. $\boldsymbol{H}_i^j = \frac{\partial^2 \boldsymbol{f}}{\partial \boldsymbol{x}_j \partial \boldsymbol{y}_i}$. The Hessian w.r.t the same variable, $\frac{\partial^2 \boldsymbol{f}}{\partial \boldsymbol{x} \partial \boldsymbol{x}^\intercal}$, can be indicated also with $\frac{\partial^2 \boldsymbol{f}}{\partial \boldsymbol{x}^2}$. For a generic function of one argument $f(\cdot)$, its derivative with respect its argument can be denoted by f'; if the argument is the time, then the derivative can be indicated also by $\dot{f}$.

Reference systems

The orientation matrix $\boldsymbol{O}_A$ of a reference system A with respect to a reference system B is a 3×3 matrix such that given a vector $\boldsymbol{v}$, if $\boldsymbol{v}_a$, $\boldsymbol{v}_b$ are the coefficients of $\boldsymbol{v}$ respectively with respect to the basis of A and B, then the following relationship holds:

$$\boldsymbol{v}_b = \boldsymbol{O}_A \boldsymbol{v}_a \tag{1}$$

When the reference frame B to which the matrix is referred is not specified, the ground frame is implicitly considered. Moreover, if $\boldsymbol{p}_a$ are the coordinates of a point P in the reference system A, the coordinates of the same point in the reference system B are:

$$\boldsymbol{p}_b = \boldsymbol{O}_A \boldsymbol{p}_a + \begin{bmatrix} x_{a,b} \\ y_{a,b} \\ z_{a,b} \end{bmatrix} \tag{2}$$

where $x_{a,b}$, $y_{a,b}$, $z_{a,b}$ are the coordinates of the origin of A with respect to B.

The orientation and the position of a reference system A with respect to a second reference system B can be jointly expressed by a 4×4 matrix $\boldsymbol{W}_A$:

$$\boldsymbol{W}_A = \left[\begin{array}{ccc|c} & & & x_{a,b} \\ & \boldsymbol{O}_A & & y_{a,b} \\ & & & z_{a,b} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \tag{3}$$

where $\boldsymbol{O}_A$ is the orientation matrix described above. The advantage of the 4×4 matrix convention is that it expresses both the rotation and translation with the same matrix, i.e. equation (2) becomes:

$$\boldsymbol{p}_b = \boldsymbol{W}_A \boldsymbol{p}_a \tag{4}$$

Rotation matrices about Cartesian axes are denoted with $\boldsymbol{R}_x$,$\boldsymbol{R}_y$ and $\boldsymbol{R}_z$:

$$\begin{gathered} \boldsymbol{R}_x(a) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos a & -\sin a & 0 \\ 0 & \sin a & \cos a & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \boldsymbol{R}_y(a) = \begin{bmatrix} \cos a & 0 & \sin a & 0 \\ 0 & 1 & 0 & 0 \\ -\sin a & 0 & \cos a & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ \boldsymbol{R}_z(a) = \begin{bmatrix} \cos a & -\sin a & 0 & 0 \\ \sin a & \cos a & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{gathered} \tag{5}$$

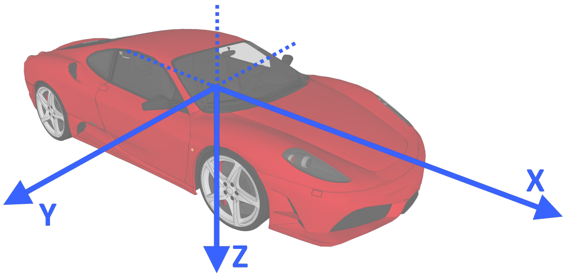


Figure 1: SAE axes convention.

Translation matrices are denoted with $\boldsymbol{T}$:

$$\boldsymbol{T}(x, y, z) = \begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6}$$

Vehicle models

SAE axes convention is adopted for the vehicle models, i.e. the frame attached to the chassis of the vehicle has the x-axis pointing to the front of the vehicle, the y-axis pointing to the right of the driver and the z-axis pointing under the road plane, as shown in figure 1.

A list of common variables used for road and vehicle models are reported in table 1.

Table 1: List of common symbols

Symbol	Description
Road variables	
x_r, y_r, z_r	road middle lane coordinates
ψ_r	road heading angle
σ_r	road slope angle
β_r	road banking angle
κ	road curvature in the $x - y$ plane
ν	road curvature in the $x - z$ plane
τ	road curvature in the $y - z$ plane (torsion)
$\boldsymbol{W}_r$	road reference frame

Symbol	Description
Vehicle and chassis variables	
s	chassis position along the road curvilinear abscissa
n	chassis lateral displacement w.r.t the road middle lane
z	chassis vertical displacement w.r.t. the nominal position
α	chassis heading w.r.t the road middle lane
ϕ	chassis roll angle w.r.t the road plane
μ	chassis pitch angle w.r.t the road plane
$\boldsymbol{W}_v$	vehicle (chassis) reference frame
$\boldsymbol{W}_{rel}$	vehicle (chassis) reference frame relative to the road one
V	vehicle speed
λ	vehicle sideslip angle
Ω	vehicle angular velocity about the axis
	perpendicular to the road surface
m	total mass
I_xx	vehicle inertia moment about chassis x-axis
I_yy	vehicle inertia moment about chassis y-axis
I_zz	vehicle inertia moment about chassis z-axis
I_xz	vehicle cross inertia moment in the $x-z$-plane
w	wheelbase
b	CoM horizontal distance from the rear axle
a	CoM horizontal distance from the front axle
h	chassis CoM nominal distance w.r.t. the road plane
t_v	vehicle half width
$t_v^{(f)}$	vehicle front axle width
$t_v^{(r)}$	vehicle rear axle width
Wheels and suspensions	
$z_w^{(\hat{a}\hat{s})1}$	wheel plate vertical displacement w.r.t. the chassis
$\omega_w^{(\hat{a}\hat{s})1}$	wheel spin velocity
δ	steering angle
$m_w^{(\hat{a})1}$	wheel mass
$I_w^{(\hat{a}\hat{s})1}$	wheel spin inertia moment
$F_s^{(\hat{a}\hat{s})1}$	suspension force

Symbol	Description
$K_{ss}^{(\hat{a})1}$	suspension spring stiffness (reduced at wheel)
$K_{sd}^{(\hat{a})1}$	suspension damper stiffness (reduced at wheel)
$K_{a}^{(\hat{a})1}$	suspension anti-roll bar stiffness (reduced at wheel)
Tyres	
$\xi_{t}^{(\hat{a}\hat{s})1}$	tyre radial deformation
$\kappa_{t}^{(\hat{a}\hat{s})1}$	tyre longitudinal slip
$\lambda_{t}^{(\hat{a}\hat{s})1}$	tyre sideslip angle
forces	
$N^{(\hat{a}\hat{s})1}$	tyre load
$F_{tx}^{(\hat{a}\hat{s})1}$	tyre longitudinal force
$F_{ty}^{(\hat{a}\hat{s})1}$	tyre lateral force
τ_e	engine torque
τ_b	braking torque
γ	front braking bias
$\tau_{wd}^{(\hat{a}\hat{s})1}$	driving torque applied to the wheel
τ_d	overall driving torque ($\tau_d = \sum_{as} \tau_{wd}^{(\hat{a}\hat{s})}$)
F_d	drag force
F_l	down-force (for cars) or lift-force (for motorbikes)
$R_{t}^{(\hat{a}\hat{s})1}$	tyre rolling resistance (force)
$\tau_{R}^{(\hat{a}\hat{s})1}$	tyre rolling resistance (torque)
$K_{tR}^{(\hat{a}\hat{s})1}$	tyre rolling resistance (force) coefficient
$K_{\tau_R}^{(\hat{a}\hat{s})1}$	tyre rolling resistance (torque) coefficient
$r_{tr}^{(\hat{a}\hat{s})1}$	tyre rolling radius
$r_{t}^{(\hat{a}\hat{s})1}$	tyre radius
$\rho_{t}^{(\hat{a}\hat{s})1}$	tyre cross section toroid radius
$K_{t\kappa}^{(\hat{a})1}$	tyre longitudinal stiffness
$K_{t\lambda}^{(\hat{a})1}$	tyre lateral stiffness
$K_{t\phi}^{(\hat{a})1}$	tyre roll stiffness
$K_{tr}^{(\hat{a})1}$	tyre radial stiffness
$K_{td}^{(\hat{a})1}$	tyre radial damping
$\mu_{x}^{(\hat{a})1}$	tyre maximum longitudinal adherence
$\mu_{y}^{(\hat{a})1}$	tyre maximum lateral adherence

Symbol	Description
Motor and powertrain	
τ_d	overall driving torque (traction and braking) acting on the wheels
τ_b	overall braking torque acting on the wheels
τ_t	overall traction torque acting on the wheels
$\tau^{(\hat{a}\hat{s})}$[1]	torque acting on a single wheel
τ_g	motor to rear axle gear ratio
k_d	differential stiffness

[1] The superscript $\hat{a} \in \{f, r\}$ indicates the front ($\hat{a} = f$) or rear ($\hat{a} = r$) axle, and $\hat{s} \in \{r, l\}$,right ($\hat{s} = r$) indicates the right or left ($\hat{s} = l$) side.

Acronyms

A list of acronyms used in this thesis is reported in table 2.

Table 2: List of acronyms

DAE	Differential Algebraic Equation
DO	Dynamic Optimisation
DP	Dynamic Programming
FJ	Fritz-John
GA	Genetic Algorithm
KKT	Karush-Kuhn-Tucker
LP	Linear Programming
LTS	Lap Time Simulation
NLP	Nonlinear Programming
OC	Optimal Control
OCP	Optimal Control Problem
PMP	Pontryagin Maximum Principle
QP	Quadratic Programming
SQP	Sequential Quadratic Programming

Chapter 1

Introduction

Optimisation problems have fascinated mathematicians since the early civilisations; in the ancient Greece, Euclid (300 BC) proved that among the rectangles with a given perimeter, the one with the greatest area is the square, and Zenodorus (200 BC) studied the similar Dido's Problem, consisting in finding the figure bounded partially by a straight line which has the maximum area for a given perimeter. Later, since the 17^{th} century, optimisation started being systematically studied by famous mathematicians. Optimal control problems (OCP) are none other than constrained optimisation problems dealing with dynamical systems; mathematically an OCP consists in the minimisation of a certain functional, constrained by the equations describing the evolution of the dynamical system and eventually by other additional equations. Optimal control theory gained of renovated interest in the second half of the 20^{th} century, and since then experienced a wide diffusion thanks to the contemporary improvements in silicon-based calculators that allowed to numerically solve a larger number of OCPs. Optimisation of road vehicles then became one of the several application fields of optimal control theory, specifically to calculate the minimum lap time of vehicles on race tracks and to optimise their design.

This thesis is focused on optimal control problems, in particular with applications to vehicle design optimisation and lap time simulations (LTS). The aim is twofold: first, optimal control solution methods are studied in order to understand the most suitable approach for vehicle-related OCPs and, second, to develop vehicle models that both improve the current state-of-the-art art of LTS models and still lead to OCPs solvable with existing software. Nowadays optimal control LTS are a very powerful tool for optimisation of road vehicles, yet they are somehow limited by solving capabilities of current numerical solver that require the adoption of properly simplified models. Within optimal control problems arising from LTS, there is need to overwhelm such limitations so as to improve simulation accuracy and to reduce computational times. This can be achieved in a twofold way: improving solving techniques for OCPs and properly modelling vehicle dynamics.

This thesis is divided into six chapters, each covering a slightly different topic. The first chapter gives an overview of the optimal control theory and of the possible strategies for solving this class of problems. Indirect and direct approaches will be described in detail since they are the most used techniques. All the content of the first chapter is derived from existing literature.

The contribution of this work comes in chapters from 3 to 6. Chapter 3 is dedicated to the analysis of the current most promising techniques for OCPs, the indirect and direct methods. The aim is to understand which approach is most suitable to solve vehicle related OCPs. Despite such methods have been used for decades in LTS, to the best knowledge of the author a thorough comparison of indirect and direct methods concerning LTS applications has never been performed. The two currently most used numerical solvers are chosen as representative for the indirect and direct methods, respectively the *Pins* and *GPOPS-II* software, and they are tested on a set of benchmark problems.

Chapters 4 and 5 are focused on the application of OC to vehicle minimum lap time problems, regarding respectively four-wheeled vehicles and an electric motorbike. Lap time simulations have become a tool widely used for the optimisation of vehicle design and setup especially in the last two decades, when the availability of efficient solvers and the increase of computer performance made such simulations computationally feasible. Since the beginning of the 00's, OC based lap time simulations experienced a significant development, and year after year they have been improving in terms of vehicle model complexity and simulated features. State-of-the art lap time simulations are capable of simulating Formula 1 lap times, taking into considerations for trim-dependent aerodynamic forces, kinetic energy recovery and secondary energy generation systems, and three dimensional roads. Despite this, commonly used car model for OC-LTS are based on the quasi-steady-state (QSS) tyre load simplification, i.e. they assume that, for a given acceleration and speed, tyre loads are the ones resulting from the stationary car trim for the same acceleration and speed. This assumption allows to significantly reduce the complexity and size of the resulting OCP, thus relieving numerical resolvability issues. However it is not completely clear how and how much the QSS tyre load simplification can affect the simulation outcomes, nor which level of complexity in the vehicle model should be adopted to get the best compromise between reliable results and short computational times. The first section of chapter 4 tries to give answer to this question. Three car models with a different level of detail in the car dynamics (and thus, different in complexity) are compared when used for lap time simulations and design optimisation. In the second section of the same chapter, a novel GP2 car model for LTS is developed. The GP2 model aims to improve the state-of-the-art of car models for LTS by abandoning the QSS tyre load assumption, so as to lead to more accurate simulation outcomes. Tyre loads are dynamically calculated from tyre radial deformations, and the suspension kinematics is included in the model, i.e. the complete roto-translation of the wheel plates is taken into considerations. Ride heights dependent aerodynamic forces and fully a coupled tyre model are included too. The GP2 model is validated through comparison with the telemetry data acquired during a qualifying lap on the Montmelò circuit in the 2012 GP2 season. The third and last section of chapter 4 focuses instead on the modelling and simulation of a racing go-kart. Go-karts are four wheeled vehicle whose distinguishing features are the lack of suspensions and rear axle differential, thus their dynamics is substantially different from that of race cars. Indeed when go-karts turn, an aligning yaw torque, that opposes to the turning manoeuvre, is generated at the rear axle; for this reason they are designed to lift the rear inner wheel when turning and braking, so as to reduce the aligning torque. Differently, rear wheel driven race cars tends to lift the front (not rear) inner wheel when turning. To the best knowledge of the author, in literature there is still lack of lap time simulations for go karts. This motivated the development of a go-kart model for lap time simulations, which is validated through comparison with experimental data acquired on the "Pista Azzurra" circuit in Jesolo (IT). The simulation outcomes are then analysed in order to understand if the simulation is able to reproduce the peculiar dynamics of the go-kart, i.e. the lifting of the rear wheel and the under-steering behaviour.

Electric and hybrid-electric vehicles are continuously gaining in popularity as the automotive industry is responding to tough environmental regulatory challenges and is aiming to meet growing consumer demand for energy consumption reduction and more sustainable living. Contemporary with the wider adoption of electric road vehicles, motor-sport too has seen an electrification movement. Due to the contemporary interest in "green" vehicles, chapter 5 is dedicated to the design optimisation of an electric motorbike taking part at the Tourist Trophy Zero challenge. This competition takes place along the full $60km$ long Snaefell Mountain Course, with motorcycle racing at high speed while using only the energy stored in their batteries. It is clear that a careful optimisation of the powertrain is fundamental to succeed. The optimisation is performed thorough optimal control LTS and is dedicated to the improvement of the electric power train. Accurate simulation of the motorbike dynamics is not of primary importance here, since the long road requires a concise yet effective model. Again, experimental data acquired during a practice session at the 2015 Tourist Trophy event by the Brunel University team will be used to validate the model.

In chapter 6 the focus is moved to implicit OCPs. Usually OCPs presented in classic textbooks are always in the explicit formulation, i.e. where the first order equations describing the dynamical system — that are the equation of motion in vehicle OCPs — are explicit in the state derivatives $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \ldots, t)$. However, dynamical systems can be more generally described by implicit first order equations $\boldsymbol{f}(\dot{\boldsymbol{x}}, \boldsymbol{x}, \ldots, t) = \mathbf{0}$. Even the majority of existing software for OCPs requires the problem to be described by explicit first order equations, yet in some applications it may be very cumbersome to write the equations in such form. For this reason in chapter 6 the implicit OCP formulation is analysed; a direct full collocation transcription method to convert the implicit OCP into a Nonlinear Programming (NLP) problem using a midpoint discretisation scheme is described. Moreover the relationship between the OCP and NLP Lagrange multiplier resulting from the adopted discretisation scheme is calculated. The described OCP to NLP transcription has been implemented into a numerical software, named *Maverick*, which is currently in use in the research group at the University of Padova in which the author worked during his PhD. This numerical software is tested on a bench problem consisting in the minimum manoeuvre time for a motorcycle running on a three-dimensional road.

General conclusions for this thesis will be drawn in chapter 7, together with some considerations of possible future work.

The road model that is used in all the lap time simulations performed in this work is described in appendix A. Any road model needs to be fed with real data in order to be used in simulations; a procedure for the calculation of the road model data from experimental images or GPS points is presented in appendix B. Finally, the motorcycle model used for the bench problem of the *Maverick* software is described in appendix C.

Chapter 2

Optimal control

In this chapter the fascinating theory of optimal control is introduced. Optimal control solving strategies based on calculus of variations are first described, then they are extended to the Pontryagin Maximum Principle. Approaches based on nonlinear programming are discussed too.

2.1 Introduction

The theory of optimal control (also known as dynamic optimisation) can be considered the natural extension of static optimisation. Static optimisation is used to find the minimum, or maximum, of a scalar function, that is defined over a finite set of real or integer numbers. Examples of static optimisation are Linear Programming (LP), Quadratic programming (QP), Nonlinear Programming (NLP), and integer, or mixed-integer, programming. In Dynamic Optimisation (DO) the target, that is the object of which a maximum or minimum is sought, is a functional, i.e. the unknowns of the optimisation problem are functions, not numbers as in static optimisation. Systematic research in the field of dynamic optimisation has been carried on since Bernoulli, Euler and Lagrange [52]. Bernoulli is the author of the famous *Brachistochrone problem*[1] [20], Euler gave birth to the Euler-Lagrange equation, a milestone of classical mechanics that exhibits a close connection to DO, Lagrange developed the well known theory of Lagrange Multipliers that is widely used also for static optimisation.

In the XX century DO experienced a great gain in interest mainly due to the necessity to give answer to practical problems. In particular, in the period after the Second World War two of the war leading countries, USA and URSS, pushed the research in DO for

[1] The *Brachistochrone problem* consists in finding the shape of the curve down which a bead sliding from rest and accelerated by gravity will slip (without friction) from one point to another in the least time.

tactical interests; the *minimum-time-to-climb* or the *minimum-intercepting-time* are two examples of problems that they were trying to solve. The former consists in finding the flight trajectory that allows a given aircraft to reach a certain altitude and speed in the minimum time; clearly, this problem was originated by the necessity to make an aircraft take off and become a threat for the enemies in the minimum time. The *minimum-intercepting-time* problem too deals with war-related purposes; it consists in finding the trajectory that a missile should follow in order to intercept an aircraft in the minimum time. The research achievements obtained in those years gave birth to the modern optimal control theory, which is still the key element for OCPs solving techniques currently used. The URSS research group, led by Lev Pontryagin, gave birth in 1959 to the famous *Pontryagin's Maximum Principle* (PMP) [76], a milestone in the optimal control theory on which indirect optimal control methods rely (as it will explained in the following sections). On the other side, Richard E. Bellman within the RAND (Research And Development) research group in Santa Monica, developed the theory of Dynamic Programming, a different approach than the PMP to calculate the solutions of OCPs. Since then OCP theory started being used for many different purposes, such as planning rocket, spaceship or airplanes trajectories, chemical reaction control, road vehicles design optimisation [82] and controlling industrial robots. A survey of classical problems that can be solved with OCP theory can be found in [11]. The enormous progress of silicon-based calculators in the second half of the 20^{th} century made the computation of numerical OCP solutions more affordable, leading to significant diffusion of OCP techniques.

In the following sections the mathematical formulation of an optimal control problem is presented, then, the two most used numerical solution methods, indirect and direct methods, will be described.

2.2 Optimal control problems

Optimal control problems, as the name suggests, are a wide class of problems where the optimal control law for a dynamical systems has to be found; a control law is optimal if it minimises (or maximises) a certain cost functional named target. The dynamical system is characterised by state variables $\boldsymbol{x}(\zeta)$, controls $\boldsymbol{u}(\zeta)$ and parameters $\boldsymbol{p}$:

$$\begin{aligned} \boldsymbol{x}(\zeta) &= \big[x_1(\zeta); \dots ; x_{n_x}(\zeta)\big] \\ \boldsymbol{u}(\zeta) &= \big[u_1(\zeta); \dots ; u_{n_u}(\zeta)\big] \\ \boldsymbol{p} &= \big[p_1; \dots ; p_{n_p}\big] \end{aligned} \tag{2.1}$$

The states and controls are function of an independent variable ζ that may be the time, however it can be more generally whatever independent variable; for instance, in the problems described in chapters 4 and 5, ζ represents the curvilinear abscissa along the circuit. The evolution of the states $\boldsymbol{x}(\zeta)$ is described by a set of first order differential equations that depends on the controls and parameters:

$$\frac{d\boldsymbol{x}(\zeta)}{d\zeta} = \boldsymbol{f}\left(\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta), \boldsymbol{p}, \zeta\right) \tag{2.2}$$

where

$$\boldsymbol{f} = [f_1; \ldots ; f_{n_x}] \tag{2.3}$$

Given an initial point $\boldsymbol{x}(\zeta_i)$, the parameters $\boldsymbol{p}$ and a control law $\boldsymbol{u}(\zeta)$, with $\zeta \in (\zeta_i,\ \zeta_f)$, the evolution of the dynamical system $\boldsymbol{x}(\zeta)$ is fully determined and can be calculated by integrating the differential equations (2.2). In optimal control problems, among all the control laws $\boldsymbol{u}(\zeta)$, the one that minimises the cost function J is sought. Moreover, the control has to belong to the so-called *admissible control set $\mathcal{U}$*. The cost functional (or target) may depend both on the value of the state variables $\boldsymbol{x}$ at the domain boundaries and on the integral value of a function ℓ:

$$J = \psi(\boldsymbol{x}(\zeta_i), \boldsymbol{x}(\zeta_f), \boldsymbol{p}, \zeta_i, \zeta_f) + \int_{\zeta_i}^{\zeta_f} \ell\left(\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta), \boldsymbol{p}, \zeta\right) d\zeta \tag{2.4}$$

where the term ψ that depends on the state at the boundaries of the independent variable is called *Mayer* target and the integral one ℓ is called *Lagrange* target.

While the dynamical system evolves from ζ_i to ζ_f, it is required to satisfy some constraints. Constraints may be in the form of boundary conditions, path constraints (i.e. constraints that must be satisfied for all $\zeta \in (\zeta_i,\ \zeta_f)$) or integral constraints:

$$\begin{aligned}
&\boldsymbol{\phi}_l \leq \boldsymbol{\phi}(\boldsymbol{x}(\zeta_i), \boldsymbol{x}(\zeta_f), \boldsymbol{p}, \zeta_i, \zeta_f) && \leq \boldsymbol{\phi}_u \\
&\boldsymbol{g}_l \leq \boldsymbol{g}(\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta), \boldsymbol{p}, \zeta) && \leq \boldsymbol{g}_u \qquad \forall \zeta \ \in\ [\zeta_i,\ \zeta_f] \\
&\boldsymbol{h}_l \leq \int_{\zeta_i}^{\zeta_f} \boldsymbol{h}(\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta), \boldsymbol{p}, \zeta) d\zeta && \leq \boldsymbol{h}_u
\end{aligned} \tag{2.5}$$

where $\boldsymbol{\phi}$ are the boundary conditions, $\boldsymbol{g}$ are the path constraints and $\boldsymbol{h}$ are the integral constraints. Summarising, the optimal control problem can be formulated as follows:

$$\min_{\boldsymbol{u}\in\mathcal{U}} \psi(\boldsymbol{x}(\zeta_i), \boldsymbol{x}(\zeta_f), \boldsymbol{p}, \zeta_i, \zeta_f) + \int_{\zeta_i}^{\zeta_f} \ell(\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta), \boldsymbol{p}, \zeta) d\zeta \tag{2.6a}$$

subject to: (2.6b)

$$\frac{d\boldsymbol{x}(\zeta)}{d\zeta} = \boldsymbol{f}(\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta), \boldsymbol{p}, \zeta) \tag{2.6c}$$

$$\boldsymbol{\phi}_l \leq \boldsymbol{\phi}(\boldsymbol{x}(\zeta_i), \boldsymbol{x}(\zeta_f), \boldsymbol{p}, \zeta_i, \zeta_f) \leq \boldsymbol{\phi}_u \tag{2.6d}$$

$$\boldsymbol{g}_l \leq \boldsymbol{g}(\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta), \boldsymbol{p}, \zeta) \leq \boldsymbol{g}_u \tag{2.6e}$$

$$\boldsymbol{h}_l \leq \int_{\zeta_i}^{\zeta_f} \boldsymbol{h}(\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta), \boldsymbol{p}, \zeta) d\zeta \leq \boldsymbol{h}_u \tag{2.6f}$$

The optimal control problem in (2.6) is the classical OCP formulation that can be found in classic textbooks as [76, 20, 75, 11]. Solution methods for problems as in (2.6) have been widely studied in the last years since a large number of optimisation problems for dynamical systems falls into this formulation. A description of solution strategies is given in the next section.

Despite being relatively general, the formulation in (2.6) presents some limitations that may prevent it from being used in certain applications. The dynamics equations (2.6c) are written in the explicit form, yet the equations for certain dynamical system are cumbersome to be written in such form. Moreover, if input controls $\boldsymbol{u}$ are discrete (integer numbers), formulation (2.6) is not directly usable[2]. While the discussion of integer or mixed-integer OCPs is out of the scope of this thesis, the formulation with implicit dynamics equations will be discussed in chapter 6.

2.3 Optimal control solving methods

In this section the solving techniques for optimal control problems are described. After a brief general introduction, the two most used methods, indirect and direct ones, are presented more in details.

Solution methods for optimal control problems can be divided into four main categories: indirect methods, direct methods, Dynamic Programming and Evolutionary Algorithms. Indirect methods rely on the Pontryagin Maximum Principle (PMP) that allows to derive the necessary conditions for the optimality of the OCP. The necessary conditions are generally in the form of a first order differential equation system with mixed initial and final boundary conditions (named Two Point Boundary Value Problem, TPBVP), plus a small

[2] If some input controls are discrete, there exists some mathematical “tricks”that allows to re-formulate the problem as in (2.6) by adding extra variables and constraints.

optimisation subproblem whose solution gives the optimal control law. When the optimisation subproblem has an explicit solution, then the controls can be substituted into the TPBVP, which can then be solved using numerical techniques for differential equations. The workflow of the indirect methods is commonly referred to as “first optimise, then discretise”, meaning that first the optimisation process is performed (necessary conditions) and then the discretisation is applied (numerical methods to discretise and solve the necessary conditions). In the direct methods, the workflow is the opposite, “first discretise, then optimise”: the OCP is first discretised accordingly to a certain discretisation scheme, and then the optimisation is process is performed on the finite dimensional algebraic variables resulting from the discretisation. A detailed description of the indirect and direct methods will be given in sections 2.3.1 and 2.3.2.

Dynamic Programming (DP) is an optimisation algorithm developed in the fifties by Richard Bellman [5] and that is based on the Hamilton-Jacobi-Bellman equation[3]. From a theoretic point of view DP is fascinating because it allows to find the optimal controls as a function of the states. This means that the optimal control law is known not only along the optimal solution, but for all the possible configurations of the dynamical system; in different words, DP gives a control feedback law. This is a great advantage when OCP are practically used to control real systems. The drawback of DP is that it is affected by the so called *curse of dimensionality*, as Bellman itself called it, that lead the DP problem to increase in size exponentially with the dimension of the state space. Thus, DP is practically feasible only for small problems. When a control feedback law is sought but DP results to be non feasible, sensitivity analysis applied to DM and IM can be used to find a feedback which is effective for small perturbations from the optimal trajectory. A thorough description of DP and/or Optimal control sensitivity analysis is out of the scope of this thesis. A brief comparison between the performance of DP, IM and DM can be found in [12, 86]; in particular, in [12] it turns out that DP algorithm is approximately 1000 times slower than IM or DM even with a problem with only two states. To mitigate the limitations of DP the Differential Dynamic Programming (DDP) was introduced that solves a sequence of quadratic subproblems obtained from the quadratic approximations of the objective function around a reference trajectory [66, 41]. The DDP was further developed by many other authors to handle highly non linear dynamics subject to state and control constrains. The Hybrid DDP is currently the state of art of DDP algorithm

[3] The Hamilton-Jacobi-Bellman equation for a dynamical system described by the first order equations $\boldsymbol{x}' = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, \zeta)$ and with a cost function $M(\boldsymbol{x}(\zeta_f)) + \int_{\zeta_i}^{\zeta_f} l(\boldsymbol{x}, \boldsymbol{u}, s)ds$, allows to calculate the *cost-to-go* function $V(\boldsymbol{x}, \zeta)$, i.e. to the optimal cost for a system starting form point $\boldsymbol{x}$ at ζ. The Hamilton-Jacobi-Bellman equation can be formulated as: $\frac{\partial V}{\partial \zeta} + \min_{\boldsymbol{u}}(\boldsymbol{f}' \frac{\partial V}{\partial \boldsymbol{x}} + l)$. Since the *cost-to-go* function $V(\boldsymbol{x}, \zeta)$ is defined over the entire state space and is not a function of the time only, it is clear the the size of DP problems increases exponentially with the state space dimension.

that combines DDP with some well-proven nonlinear mathematical programming techniques and was successfully used to solve a large scale spacecraft trajectory optimisation problem [54, 53].

Finally, Evolutionary Algorithms (EA) are a generic class of meta-heuristic optimisation algorithms. They uses mechanisms inspired by evolution of biological species to find the optimal solution of the problem. Generally, they start with a given population of individuals in the state space and then apply mechanism such as reproduction, mutation, recombination, and selection to find individuals that make the cost function to decrease. A detailed survey can be found in [4, 32]. Due to their stochastic nature, EAs are generally significantly slow compared to indirect or direct methods; moreover they provide suboptimal solutions and no optimality condition is guaranteed. However they can be straightforwardly used with non-differentiable problems and are more suitable to find global minima.

A comprehensive summary of optimal control approaches and numerical solution methods can be found in [78, 38, 12]. In this chapter the main focus is on indirect and direct methods for OCPs since practical experience shows that they are the most suitable for a large variety of problems including optimal control of road vehicles. Indeed, as previously stated, classical DP is practically non feasible even for small problems, and, even if hybrid DDP is a promising method thanks to its robustness and large convergence radius, yet it has not been applied to challenging minimum lap time problems. On the other side, EAs represent a reasonable approach only for those problems that cannot be solved with DM or IM, being EAs much more demanding (in terms of computing power) than DM or IM. In the next sections, indirect and direct methods are described more in detail.

2.3.1 Indirect methods

Indirect methods make use of the calculus of variations (CV) or the Pontryagin Maximum Principle (PMP) to derive the OCP first-order necessary conditions, and then the optimal control is calculated solving the necessary conditions. The PMP can be considered as an improvement over the calculus of variations (CV) since it allows to solve a wider family of problems. In the next subsections, CV is used to derive the necessary conditions of a case-of-study optimal control problem; then the PMP is presented and compared to CV.The penalty approach, that allows to handle inequality constraints with the indirect approach, is then described. Finally this section will briefly introduce numerical methods used to solve the two points boundary value problems that arise from IMs.

2.3.1.1 Calculus of variations

The theory of calculus of variations is known since Euler and Lagrange, indeed the Euler-Lagrange equations for classical mechanics are obtained by CV applied to the *Principle of Least (or Stationary) Action* [52]. The calculus of variations, that relies on the Lagrange multiplier method, can be applied to OCPs that do not include any inequality constraint. Consider now the following optimal control problem, obtained from (2.6) by neglecting the inequality constraints and letting the admissible controls be all continuous functions from $\mathbb{R}$ to $\mathbb{R}^{n_u}$:

$$\min_{\boldsymbol{u},\zeta_i,\zeta_f} \; \psi(\boldsymbol{x}(\zeta_i), \boldsymbol{x}(\zeta_f), \boldsymbol{u}(\zeta_i), \boldsymbol{u}(\zeta_f), \boldsymbol{p}, \zeta_i, \zeta_f) + \int_{\zeta_i}^{\zeta_f} \ell(\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta), \boldsymbol{p}, \zeta) d\zeta \tag{2.7a}$$

$$\text{subject to:} \tag{2.7b}$$

$$\frac{d\boldsymbol{x}(\zeta)}{d\zeta} = \boldsymbol{f}(\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta), \boldsymbol{p}, \zeta) \tag{2.7c}$$

$$\boldsymbol{g}(\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta), \boldsymbol{p}, \zeta) = \boldsymbol{0} \tag{2.7d}$$

$$\boldsymbol{\phi}(\boldsymbol{x}(\zeta_i), \boldsymbol{x}(\zeta_f), \boldsymbol{p}, \zeta_i, \zeta_f) = \boldsymbol{0} \tag{2.7e}$$

$$\int_{\zeta_i}^{\zeta_f} \boldsymbol{h}(\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta), \boldsymbol{p}, \zeta) d\zeta = \boldsymbol{h}^* \tag{2.7f}$$

In particular, the path $\boldsymbol{g}$ and integral $\boldsymbol{h}$ constraints are here considered as equality constraints. Using the Lagrange multipliers method, the above presented problem (2.7) can be converted into an equivalent unconstrained minimisation one with some auxiliary variables, i.e. the Lagrange multipliers (sometimes referred to also as adjoint variables). The number of Lagrange multipliers is equal to the number of constraints, and the resulting unconstrained problem is:

$$\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{u},\zeta_i,\zeta_f,\boldsymbol{\gamma},\boldsymbol{\lambda},\boldsymbol{\rho},\boldsymbol{\mu}} \; \hat{J} =& \boldsymbol{\psi}(\boldsymbol{x}(\zeta_i), \boldsymbol{x}(\zeta_f), \boldsymbol{u}(\zeta_i), \boldsymbol{u}(\zeta_f), \boldsymbol{p}, \zeta_i, \zeta_f) \\
&+ \boldsymbol{\gamma}^{\mathsf{T}} \boldsymbol{\phi}(\boldsymbol{x}(\zeta_i), \boldsymbol{x}(\zeta_f), \boldsymbol{p}, \zeta_i, \zeta_f) \\
&+ \int_{\zeta_i}^{\zeta_f} \ell(\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta), \boldsymbol{p}, \zeta) + \boldsymbol{\lambda}(\zeta)^{\mathsf{T}} \left(\frac{d\boldsymbol{x}(\zeta)}{d\zeta} - \boldsymbol{f}(\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta), \boldsymbol{p}, \zeta) \right) \\
&+ \boldsymbol{\rho}(\zeta)^{\mathsf{T}} \boldsymbol{g}(\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta), \boldsymbol{p}, \zeta) + \boldsymbol{\mu}^{\mathsf{T}} \left(\boldsymbol{h}(\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta), \boldsymbol{p}, \zeta) - \boldsymbol{h}^* \right) d\zeta
\end{aligned} \tag{2.8}$$

where $\boldsymbol{\lambda}(\zeta)$ and $\boldsymbol{\rho}(\zeta)$ are the ζ-dependent multipliers associated respectively to the dynamics equations and path constraints, $\boldsymbol{\psi}$ and $\boldsymbol{\mu}$ are the ζ-independent multipliers associated respectively to the boundary conditions and integral constraints.

According to the Lagrange multiplier method, under the assumption that the functions in (2.8) have continuous first order derivative, if J^* is the minimum for J in (2.7) and

$\boldsymbol{u}(\zeta) = \boldsymbol{u}^*(\zeta)$, $\zeta_i = \zeta_i^*$, $\zeta_f = \zeta_f^*$ are the minimisers, then the minimum for problem (2.8) is still J^*, and it is obtained for the same minimisers $\boldsymbol{u}(\zeta) = \boldsymbol{u}^*(\zeta)$, $\zeta_i = \zeta_i^*$ and $\zeta_f = \zeta_f^*$. Therefore the search for a minimiser of the constrained problem (2.7) is reduced to the search for a minimizer of the unconstrained problem (2.8).

The minimum for (2.8) can now be calculated through the calculus of variation, which allows to obtain the first order necessary conditions for the minimum. Indeed, since a minimum is a stationary point, the first order variation of J' must be zero at the minimum. The first order variation of $\hat{J}$, with respect to $\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta), \zeta_i, \zeta_f, \boldsymbol{\gamma}, \boldsymbol{\lambda}(\zeta), \boldsymbol{\rho}(\zeta), \boldsymbol{\mu}$, is:

$$\begin{aligned}
\delta\hat{J} =& \delta\zeta_i \left(\frac{\partial\psi}{\partial\zeta_i} + \boldsymbol{\gamma}^\mathsf{T} \frac{\partial\boldsymbol{\phi}}{\partial\zeta_i} - \left(\ell\Big|_{\zeta_i} + \boldsymbol{\lambda}^\mathsf{T}(\zeta_i)\boldsymbol{f}\Big|_{\zeta_i} + \boldsymbol{\rho}^\mathsf{T}(\zeta_i)\boldsymbol{g}\Big|_{\zeta_i} + \boldsymbol{\mu}^\mathsf{T}\boldsymbol{h}\Big|_{\zeta_i} \right) \right) && \text{(2.9a)}\\
&+\delta\zeta_f \left(\frac{\partial\psi}{\partial\zeta_f} + \boldsymbol{\gamma}^\mathsf{T} \frac{\partial\boldsymbol{\phi}}{\partial\zeta_f} + \left(\ell\Big|_{\zeta_f} + \boldsymbol{\lambda}^\mathsf{T}(\zeta_f)\boldsymbol{f}\Big|_{\zeta_f} + \boldsymbol{\rho}^\mathsf{T}(\zeta_f)\boldsymbol{g}\Big|_{\zeta_f} + \boldsymbol{\mu}^\mathsf{T}\boldsymbol{h}\Big|_{\zeta_f} \right) \right) && \text{(2.9b)}\\
&+\delta\boldsymbol{x}(\zeta_i)^\mathsf{T} \left(\frac{\partial\psi}{\partial\boldsymbol{x}(\zeta_i)} + \boldsymbol{\gamma}^\mathsf{T} \frac{\partial\boldsymbol{\phi}}{\partial\boldsymbol{x}(\zeta_i)} + \boldsymbol{\lambda}^\mathsf{T}(\zeta_i) \right) && \text{(2.9c)}\\
&+\delta\boldsymbol{x}(\zeta_f)^\mathsf{T} \left(\frac{\partial\psi}{\partial\boldsymbol{x}(\zeta_f)} + \boldsymbol{\gamma}^\mathsf{T} \frac{\partial\boldsymbol{\phi}}{\partial\boldsymbol{x}(\zeta_f)} - \boldsymbol{\lambda}^\mathsf{T}(\zeta_f) \right) && \text{(2.9d)}\\
&+\int_{\zeta_i}^{\zeta_f} \delta\boldsymbol{x}^\mathsf{T} \left(\frac{\partial\ell}{\partial\boldsymbol{x}} + \boldsymbol{\lambda}^\mathsf{T}\frac{\partial\boldsymbol{f}}{\partial\boldsymbol{x}} + \boldsymbol{\rho}^\mathsf{T}\frac{\partial\boldsymbol{g}}{\partial\boldsymbol{x}} + \boldsymbol{\mu}^\mathsf{T}\frac{\partial\boldsymbol{h}}{\partial\boldsymbol{x}} \right) d\zeta && \text{(2.9e)}\\
&+\int_{\zeta_i}^{\zeta_f} \delta\boldsymbol{u}^\mathsf{T} \left(\frac{\partial\ell}{\partial\boldsymbol{u}} + \boldsymbol{\lambda}^\mathsf{T}\frac{\partial\boldsymbol{f}}{\partial\boldsymbol{u}} + \boldsymbol{\rho}^\mathsf{T}\frac{\partial\boldsymbol{g}}{\partial\boldsymbol{u}} + \boldsymbol{\mu}^\mathsf{T}\frac{\partial\boldsymbol{h}}{\partial\boldsymbol{u}} \right) d\zeta && \text{(2.9f)}\\
&+\delta\boldsymbol{p}^\mathsf{T} \left(\frac{\partial\psi}{\partial\boldsymbol{p}} + \boldsymbol{\gamma}^\mathsf{T}\frac{\partial\boldsymbol{\phi}}{\partial\boldsymbol{p}} + \int_{\zeta_i}^{\zeta_f} \left(\frac{\partial\ell}{\partial\boldsymbol{p}} + \boldsymbol{\lambda}^\mathsf{T}\frac{\partial\boldsymbol{f}}{\partial\boldsymbol{p}} + \boldsymbol{\mu}^\mathsf{T}\frac{\partial\boldsymbol{h}}{\partial\boldsymbol{p}} \right) d\zeta \right) && \text{(2.9g)}\\
&+\delta\boldsymbol{\gamma}^\mathsf{T}\boldsymbol{\phi} + \delta\boldsymbol{\mu}^\mathsf{T} \left(\int_{\zeta_i}^{\zeta_f} \boldsymbol{h}\ d\zeta - \boldsymbol{h}^* \right) + \int_{\zeta_i}^{\zeta_f} \delta\boldsymbol{\lambda}^\mathsf{T} \left(\frac{d\boldsymbol{x}(\zeta)}{d\zeta} - \boldsymbol{f} \right)\ d\zeta && \text{(2.9h)}
\end{aligned}$$

Since at any stationary point the variation of $\hat{J}$ must vanish for any arbitrary variation of $\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta), \zeta_i, \zeta_f, \boldsymbol{\gamma}, \boldsymbol{\lambda}(\zeta), \boldsymbol{\rho}(\zeta)$ and $\boldsymbol{\mu}$, each term that multiplies $\delta\boldsymbol{x}(\zeta), \delta\boldsymbol{u}(\zeta), \delta\zeta_i, \delta\zeta_f, \delta\boldsymbol{\gamma}, \delta\boldsymbol{\lambda}, \delta\boldsymbol{\rho}$ and $\delta\boldsymbol{\mu}$ must be identically zero. Imposing that such coefficients must be zero, the necessary conditions are thus obtained:

$$\begin{aligned}
\boldsymbol{0} =& \frac{\partial\psi}{\partial\zeta_i} + \boldsymbol{\gamma}^\mathsf{T} \frac{\partial\boldsymbol{\phi}}{\partial\zeta_i} - \left(\ell\Big|_{\zeta_i} + \boldsymbol{\lambda}^\mathsf{T}(\zeta_i)\boldsymbol{f}\Big|_{\zeta_i} + \boldsymbol{\rho}^\mathsf{T}(\zeta_i)\boldsymbol{g}\Big|_{\zeta_i} + \boldsymbol{\mu}^\mathsf{T}\boldsymbol{h}\Big|_{\zeta_i} \right) && \text{(2.10a)}\\
\boldsymbol{0} =& \frac{\partial\psi}{\partial\zeta_f} + \boldsymbol{\gamma}^\mathsf{T} \frac{\partial\boldsymbol{\phi}}{\partial\zeta_f} + \left(\ell\Big|_{\zeta_f} + \boldsymbol{\lambda}^\mathsf{T}(\zeta_f)\boldsymbol{f}\Big|_{\zeta_f} + \boldsymbol{\rho}^\mathsf{T}(\zeta_f)\boldsymbol{g}\Big|_{\zeta_f} + \boldsymbol{\mu}^\mathsf{T}\boldsymbol{h}\Big|_{\zeta_f} \right) && \text{(2.10b)}\\
\boldsymbol{0} =& \frac{\partial\psi}{\partial\boldsymbol{x}(\zeta_i)} + \boldsymbol{\gamma}^\mathsf{T} \frac{\partial\boldsymbol{\phi}}{\partial\boldsymbol{x}(\zeta_i)} + \boldsymbol{\lambda}^\mathsf{T}(\zeta_i) && \text{(2.10c)}
\end{aligned}$$

$$\mathbf{0} = \frac{\partial \psi}{\partial \boldsymbol{x}(\zeta_f)} + \boldsymbol{\gamma}^\mathsf{T} \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{x}(\zeta_f)} - \boldsymbol{\lambda}^\mathsf{T}(\zeta_f) \tag{2.10d}$$

$$\mathbf{0} = \frac{\partial \ell}{\partial \boldsymbol{x}} + \boldsymbol{\lambda}^\mathsf{T} \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}} + \boldsymbol{\rho}^\mathsf{T} \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}} + \boldsymbol{\mu}^\mathsf{T} \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}} \tag{2.10e}$$

$$\mathbf{0} = \frac{\partial \ell}{\partial \boldsymbol{u}} + \boldsymbol{\lambda}^\mathsf{T} \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}} + \boldsymbol{\rho}^\mathsf{T} \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{u}} + \boldsymbol{\mu}^\mathsf{T} \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{u}} \tag{2.10f}$$

$$\mathbf{0} = \frac{\partial \psi}{\partial \boldsymbol{p}} + \boldsymbol{\gamma}^\mathsf{T} \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{p}} + \int_{\zeta_i}^{\zeta_f} \left(\frac{\partial \ell}{\partial \boldsymbol{p}} + \boldsymbol{\lambda}^\mathsf{T} \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{p}} + \boldsymbol{\mu}^\mathsf{T} \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{p}} \right) d\zeta \tag{2.10g}$$

$$\mathbf{0} = \boldsymbol{\phi} \tag{2.10h}$$

$$\mathbf{0} = \int_{\zeta_i}^{\zeta_f} \boldsymbol{h} \; d\zeta - \boldsymbol{h}^* \tag{2.10i}$$

$$\mathbf{0} = \frac{d\boldsymbol{x}(\zeta)}{d\zeta} - \boldsymbol{f} \tag{2.10j}$$

$$\mathbf{0} = \boldsymbol{g} \tag{2.10k}$$

Equations (2.10) represent the necessary conditions for the unconstrained minimisation problem (2.8), thus for the optimal control problem (2.7). They are called *necessary* because they are satisfied by any stationary point for the functional J', thus must be satisfied at the minimum too. However, a solution of (2.10) may be even a maximum or a saddle point. The second order variation information should be used in order to ensure that the solution is a minimum. It should be noted that the controls $\boldsymbol{u}$ satisfying the necessary conditions are continuous since all functions that appear in (2.10) have continuous first order derivatives by hypothesis.

The unknowns of problem (2.7) are the domain extrema ζ_i, ζ_f, the states $\boldsymbol{x}$, controls $\boldsymbol{u}$, parameters $\boldsymbol{p}$ together with the ζ-dependent $\boldsymbol{\lambda}$, $\boldsymbol{\rho}$ and ζ-independent $\boldsymbol{\gamma}$, $\boldsymbol{\mu}$ Lagrange multipliers. If n_x, n_u, n_p, n_g, n_h, n_ϕ are respectively the number of states, controls, parameters, path constraints, integral constraints and boundary conditions, there are $2n_x$ differential unknowns, n_u+n_g ζ-dependent algebraic unknowns and $n_p+n_h+2n_\phi$ unknown parameters (ζ-independent). Therefore the solution is fully determined by $2n_x$ differential equations with as many boundary conditions (algebraic equations) plus n_u+n_g ζ-dependent algebraic equations and n_h+2n_ϕ ζ-independent algebraic equations. Indeed, it can be noted equations (2.10) include exactly $2n_x$ differential equations (2.10j), (2.10e), with as many boundary conditions (2.10c), (2.10d), $n_u + n_g$ ζ-dependent algebraic equations (2.10f), (2.10k), and $n_p + n_h + 2n_\phi$ ζ-independent algebraic equations (2.10g), (2.10i), (2.10h), (2.10a), (2.10b).

2.3.1.2 Pontryagin Maximum Principle

The Pontryagin Maximum Principle is the result of the work made by Lev Pontryagin and his collaborators in the '50s, and it is a development over the classical theory of the Calculus of Variations. The PMP will be first described as stated by Pontryagin in his work [76], then it will be compared to the CV approach presented in the previous pages.

Let's consider a dynamical systems defined by a set of first order equations:

$$\frac{d\boldsymbol{x}(\zeta)}{d\zeta} = \boldsymbol{f}\left(\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta)\right) \tag{2.11}$$

where again $\boldsymbol{x}(\zeta)$ and $\boldsymbol{u}(\zeta)$ are respectively the states and the controls. The functions $\boldsymbol{f}(\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta), \zeta)$ are supposed to be continuously differentiable with respect to $\boldsymbol{x}$, and continuous in $\boldsymbol{u}$ and ζ. The controls $\boldsymbol{u}$ are not arbitrary functions in $\mathcal{R}^{n_u}$, but belong to the so called *admissible* controls. Admissible controls are the functions $\boldsymbol{u}$ that lay within the control region $\mathcal{U} \subset \mathcal{R}^{n_u}$, and that are piecewise continuous, i.e. $\boldsymbol{u} \in \mathcal{U}$ are continuous except at most on a finite number of points. Among all the admissible controls, the optimal one is the control that makes the system to move from $\boldsymbol{x}_i$ at $\zeta = \zeta_i$ to $\boldsymbol{x}_f$ at $\zeta = \zeta_f$ while minimising the cost function:

$$J = \int_{\zeta_i}^{\zeta_f} f_0\left(\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta)\right) \tag{2.12}$$

where f_0 is continuously differentiable with respect to $\boldsymbol{x}$, and continuous in $\boldsymbol{u}$ and ζ. The theorem stated by the PMP allows to find the necessary conditions that the optimal controls and trajectory must satisfy. It should be noted that every solution $\boldsymbol{x}(\zeta)$ of (2.11) is continuous in $[\zeta_i,\ \zeta_f]$, and its ζ-derivative is piecewise continuous (it is not continuous where the control $\boldsymbol{u}$ is not continuous).

Before the PMP theorem is presented, some basic definitions are now given.

The cost function (2.12) can be included in the state variables by adding a proper zero entry x_0 to $\boldsymbol{x}$ and f_0 to $\boldsymbol{f}$:

$$\begin{aligned} \boldsymbol{x}(\zeta) &= \Big[x_0(\zeta); \quad x_1(\zeta); \quad \ldots; \quad x_{n_x}(\zeta)\Big] \\ \boldsymbol{f}(\zeta) &= \Big[f_0\left(\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta)\right); \quad f_1\left(\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta)\right); \quad \ldots; \quad f_{n_x}\left(\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta)\right)\Big] \end{aligned} \tag{2.13}$$

with initial condition $x_{0_i} = 0$. The cost function (2.12) thus becomes:

$$J = \int_{\zeta_i}^{\zeta_f} f_0\left(\boldsymbol{x}(\zeta), \boldsymbol{u}(\zeta)\right) = x_{0_f} \tag{2.14}$$

Definition 1 *Line* Γ.
The line Γ *in the space* $\mathcal{R}^{n_x+1}$ *is a line parallel to the* x_0 *axis and passing through the point* $(0, \boldsymbol{x}_f)$.

Definition 2 *Auxiliary variables $\boldsymbol{\Psi}(\zeta)$.*
The variables $\boldsymbol{\Psi}(\zeta) = [\Psi_0(\zeta); \Psi_1(\zeta); \ldots ; \Psi_{n_x}(\zeta)]$ are called auxiliary variables. They satisfy (by definition) the first-order equations:

$$\frac{d\Psi_i}{d\zeta} = -\boldsymbol{\Psi}(\zeta)^{\mathsf{T}} \frac{\partial \boldsymbol{f}(\zeta)}{\partial x_i} \qquad i = 0, \ldots, n_x \tag{2.15}$$

It should be noted that the auxiliary variables $\boldsymbol{\Psi}$ are thus continuous in the interval $[\zeta_i,\ \zeta_f]$, and its ζ-derivative is piecewise continuous (it is not continuous where the control $\boldsymbol{u}$ is not continuous).

Definition 3 *Hamiltonian $\mathcal{H}$.*
The Hamiltonian $\mathcal{H}$ of the system (scalar function) is defined as follows:

$$\mathcal{H}\left(\boldsymbol{x}, \boldsymbol{\Psi}, \boldsymbol{u}\right) = -\boldsymbol{\Psi}^{\mathsf{T}} \boldsymbol{f}\left(\boldsymbol{x}, \boldsymbol{u}\right) \tag{2.16}$$

The Pontryagin Maximum Principle theorem can now be stated:

Theorem 1 *Pontryagin Maximum Principle - fixed endpoints*
Let $\boldsymbol{u}(\zeta)$, $\zeta_i \leq \zeta \leq \zeta_f$, be an admissible control such that corresponding trajectory $\boldsymbol{x}(\zeta)$ which begins at point $\boldsymbol{x}_0$ at ζ_i passes, at some time ζ_f through a point on the line Γ. In order that $u(\zeta)$ and $x(\zeta)$ be optimal it is necessary that there exists a nonzero continuous vector function $\boldsymbol{\Psi}(\zeta)$, corresponding to $\boldsymbol{u}(\zeta)$ and $\boldsymbol{x}(\zeta)$such that:

1. *for every $\zeta:\quad \zeta_i \leq \zeta \leq \zeta_f$ the function $\mathcal{H}(\boldsymbol{x}(\zeta), \boldsymbol{\Psi}(\zeta), \boldsymbol{u})$ attains its maximum value* [4] *at the point $\boldsymbol{u} = \boldsymbol{u}(\zeta)$:*

$$\mathcal{H}\left(\boldsymbol{x}(\zeta), \boldsymbol{\Psi}(\zeta), \boldsymbol{u}(\zeta)\right) = \sup_{\boldsymbol{u}} \mathcal{H}\left(\boldsymbol{x}(\zeta), \boldsymbol{\Psi}(\zeta), \boldsymbol{u}\right) \tag{2.17}$$

2. *at the boundary ζ_f, the following relations are satisfied:*

$$\sup_{\boldsymbol{u}} \mathcal{H}\left(\boldsymbol{x}(\zeta_f), \boldsymbol{\Psi}(\zeta_f), \boldsymbol{u}(\zeta_f)\right) = 0, \qquad \Psi_0(\zeta_f) \leq 0 \tag{2.18}$$

Furthermore, it turns out that if $\boldsymbol{x}(\zeta), \boldsymbol{\Psi}(\zeta), \boldsymbol{u}(\zeta)$ satisfy equations (2.11) *and* (2.15)*, the time functions $\Psi_0(\zeta)$ and $\sup_{\boldsymbol{u}} \mathcal{H}(\boldsymbol{x}(\zeta_f), \boldsymbol{\Psi}(\zeta_f), \boldsymbol{u}(\zeta_f))$ are constant. Thus equation* (2.18) *may be verified at any ζ, not just at ζ_f.*

The above stated theorem can be generalised to the case with variable endpoints. The problem with variable endpoints can be formulated as follows. Let S_i and S_f be two

[4] This is the reason why the Pontryagin Principle is called *Maximum* principle, even if it is used to minimise a cost functional.

manifolds in $\mathcal{R}^{n_x}$ of arbitrary (but less than n_x) dimensions, respectively r_i and r_f. The problem is to find an admissible control $u(\zeta)$ which transfers the phase point from any position $\boldsymbol{x}_i \in S_i$ to any position $\boldsymbol{x}_f \in S_f$, and which in doing so imparts a minimum value to the functional (2.12). In this case the problem has to satisfy the so called *transversality conditions*, which relate the trajectory at the boundaries of ζ with the manifolds S_i and S_f. The formulation of the transversality conditions is now given. Let $\boldsymbol{x}_i \in S_i$ and $\boldsymbol{x}_f \in S_f$ be certain points, and let T_i and T_f be the tangent planes of S_i and S_f which pass through these points. The planes T_i and T_f have dimensions r_i and r_f, respectively. Furthermore, let $\boldsymbol{u}(\zeta), \boldsymbol{x}(\zeta)$ be the solution of the optimal problem with the fixed endpoints $\boldsymbol{x}_i$ and $\boldsymbol{x}_f$. Finally, let $\boldsymbol{\Psi}(\zeta)$ be a vector whose existence is asserted in theorem (1). The vector $\boldsymbol{\Psi}(\zeta)$ satisfies the transversality conditions at the left and right endpoint of the trajectory, respectively if $\boldsymbol{\Psi}(\zeta_i)$ is orthogonal to T_i and if $\boldsymbol{\Psi}(\zeta_f)$ is orthogonal to T_f. The PMP can now be stated for the variable endpoints problem:

Theorem 2 *Pontryagin Maximum Principle - variable endpoints*
Let $\boldsymbol{u}(\zeta)$, $\zeta_i \leq \zeta \leq \zeta_f$, be an admissible control that transfers the system from an initial point $\boldsymbol{x}_i$ to a final point $\boldsymbol{x}_f$, and let $\boldsymbol{x}(\zeta)$ be the corresponding trajectory. In order that $\boldsymbol{u}(\zeta)$ and $\boldsymbol{x}(\zeta)$ yield the solution of the optimal problem with variable endpoints, it is necessary that there exists a nonzero continuous vector function $\boldsymbol{\Psi}(\zeta)$, which satisfies the conditions of theorem (1)*, and in addition, the transversality condition at both endpoints of the trajectory $\boldsymbol{x}(\zeta)$. If either S_i or S_f degenerates into a point, the transversality condition is replaced by the condition that $\boldsymbol{x}$ passes through that point.*

Theorems (1) and (2) deals with a dynamical system where the first order equations (2.11) and the cost function (2.12) do not depend explicitly on the independent variable ζ. The necessary conditions in such case can be straightforwardly derived from theorems (1) and (2) by augmenting the state variables vector $\boldsymbol{x}$ with an extra entry x_{nx+1} which is equal to ζ, i.e. $f_{nx+1} = 1$. The problem is thus reduced to the one discussed in theorems (1) and (2), and the necessary conditions can be obtained with the PMP. Integral constraints present in the OCP given in (2.7) can be converted to boundary conditions by properly adding extra state variables into the state vector $\boldsymbol{x}$. The interested reader can find more details in [76].

It is now clear how the Pontryagin Maximum Principle has a wider application field than the calculus of variations. First of all, CV requires the functions to have continuous first order derivatives, thus the resulting controls are continuous. Differently, PMP can be used to solve problems whose optimal controls have discontinuities at most on a finite number of points. Moreover PMP can deal with problems where the controls must lay within an admissible region $\mathcal{U}$, while in CV controls are assumed to span all the function space. Both

PMP and CV however does not handle problems with inequality constraints; the penalty approach can be used to overcome this limitation, as described in the next section.

2.3.1.3 Inequality constraints and penalties

In the optimal control problem solved with the CV and PMP in the previous pages, all the constraints are equality constraints, yet an OCP may in general include inequality constraints, as in (2.6). This apparently means that neither CV nor PMP can be used to solve such kind of problems. Using the penalty approach it is possible to transform the inequality constraints into penalty functions, which are included into the cost function. The inequality-constrained OCP is thus converted into an (almost) equivalent one without inequalities, that can thus be solved through CV or PMP. The penalty approach is now presented.

Consider the OCP (2.6), described in the previous section, that includes inequality constraints. Inequality constraints comprise boundary terms (2.6d), path constraints (2.6e) and integral terms (2.6f). Let's now focus only on the boundary and path inequalities, neglecting the integral constraints; the case with integral constraints will be discussed later. Penalty functions (or simply penalties) associated to these inequality constraints p_ϕ, p_g are whatever type of functions that assumes small (relatively to the OCP target) values when the constraints are satisfied, and high values when they are not:

$$
\begin{aligned}
p_\phi(\phi) &\approx \begin{cases} \epsilon_\phi \ll \psi, & \text{if } \ \phi_l \le \phi \le \phi_u \\ M_\phi \gg \psi, & \text{otherwise} \end{cases} \\
p_g(g) &\approx \begin{cases} \epsilon \ll \ell, & \text{if } \ g_l \le g \le g_u \\ M \gg \ell, & \text{otherwise} \end{cases}
\end{aligned}
\tag{2.19}
$$

Accordingly to the above described penalty form, the penalties present an abrupt variation (i.e. high derivative) when exceeding the constraint boundaries. The derivative of the penalties is the most relevant property of a penalty function than its value; indeed, penalties which differ for a constant quantity give the same OCP solution — even though they may not be equivalent from a numerical point of view. Penalties in (2.19) are general; it is left to the user (or to the OCP solver) to define a mathematical function for the penalties which satisfies (2.19). The underlying idea of the penalty approach is that, once p_ϕ and p_g have been defined accordingly to (2.19), the original optimal control problem (2.6) (without integral constraints) can be modified to an almost equivalent one by adding the penalty functions into the cost functional as follows:

$$\begin{aligned}\min_{\boldsymbol{u}}\ & \psi(\boldsymbol{x}(\zeta_i),\boldsymbol{x}(\zeta_f),\boldsymbol{u}(\zeta_i),\boldsymbol{u}(\zeta_f),\boldsymbol{p},\zeta_i,\zeta_f) \\ & + p_\phi\left(\phi\left(\boldsymbol{x}(\zeta_i),\boldsymbol{x}(\zeta_f),\boldsymbol{u}(\zeta_i),\boldsymbol{u}(\zeta_f),\boldsymbol{p},\zeta_i,\zeta_f\right)\right) \\ & + \int_{\zeta_i}^{\zeta_f} \ell(\boldsymbol{x}(\zeta),\boldsymbol{u}(\zeta),\boldsymbol{p},\zeta) + p_g\left(g\left(\boldsymbol{x}(\zeta),\boldsymbol{u}(\zeta),\boldsymbol{p},\zeta\right)\right) d\zeta\end{aligned} \tag{2.20a}$$

subject to: (2.20b)

$$\frac{d\boldsymbol{x}(\zeta)}{d\zeta} = \boldsymbol{f}\left(\boldsymbol{x}(\zeta),\boldsymbol{u}(\zeta),\boldsymbol{p},\zeta\right) \tag{2.20c}$$

The inequality constraints have thus been converted, through the penalty functions, to cost functional. The only constraints left are the differential equation for the dynamical system (2.20c). The solution of the OCP with penalty functions is expected to be close to the one of the original OCP (2.6): since the penalty functions are minimised, the inequality constraints are (very likely) satisfied. The two solutions are not exactly the same since the penalty functions act as a regularisation terms because they still affect the target functional when the constraints are satisfied. It is not guaranteed that the solution of (2.20) does not violate the inequality constraints. Indeed, if the penalty function is not enough abrupt in correspondence of the constraint bounds, the solution may violate the constraints or it may remain far within the constraints bounds. The use of particular penalty functions, like barrier penalties, can prevent the solution to violate the original constraints.

As previously said, the mathematical expression for penalties is not unique and different functions may be used. Some example of penalty functions are now given. Possible penalty functions are reported in figure 2.1, where the constraint upper and lower bounds have been normalised to -1 and 1 respectively. In the leftmost and centre plot the penalties are of barrier type, i.e. they tend to infinity when approaching the constraint boundaries; such penalties enforce the solution to satisfy the constraints. The barrier penalty in the leftmost plot is the logarithmic penalty p_{log} given by:

$$p_{log} \propto -\log\left(1-c_x^2\right); \quad c_x \equiv \frac{c-\frac{c_u+c_l}{2}}{c_u-c_l}; \tag{2.21}$$

where c is the constraint value, and c_u and c_l are the upper and lower bounds. In the rightmost plot the penalty is a polynomial. The polynomial penalty does not goes to infinity when the constraint approaches its upper or lower bound, thus the solution may exceed the constraints boundaries if the penalty weight is too weak. Polynomial penalties, together with all smooth and non-barrier penalties, have the advantage to be easier to handle numerically.

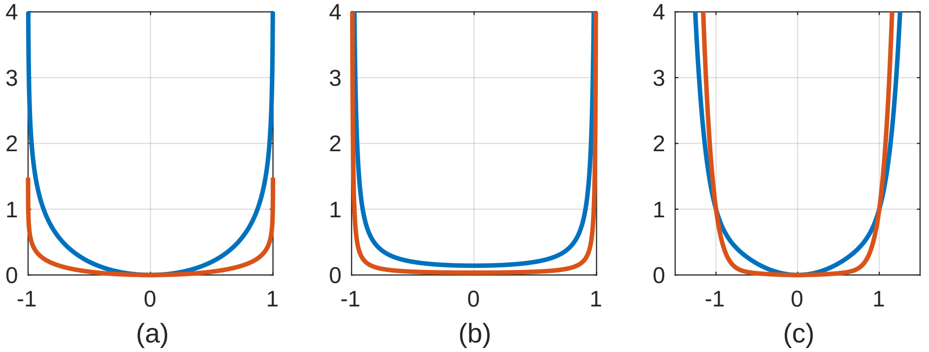


Figure 2.1: Examples of penalty function. In (a), the penalty $p(c)$ has the logarithmic expression given in equation (2.21); in (b), the penalty is proportional to $1/\cos(x\pi/2)$, while in (c) the penalty is $p(c) = hx^2 + (1-h)x^{10}$. In (a) and (b), coloured lines show different penalty weight; in (c) they refer to different value of the parameter h.

2.3.1.4 Notes on numerical methods

As seen in the previous sections, first order necessary conditions for optimal control problems (see theorem (1) or equations (2.10)) come in the form of a first order differential equations system with mixed initial and final boundary conditions (TPBVP), plus a small optimisation subproblem that gives the optimal control law. Numerical methods for indirect OCP approaches must therefore solve a system of differential equations with mixed boundary conditions; the optimisation subproblem can, in certain cases, be explicitly solved and the control are thus substituted in the equations.

Most used numerical approaches for OCP indirect methods are shooting, multiple shooting and finite discretisation. In the shooting method, an initial guess on the initial (or final) boundary conditions and control history is used to integrate forward (or backward) the differential equations, including both states and Lagrange multipliers. The resulting final (or initial) state resulting from the integration may not satisfy the final (or initial) boundary conditions. Thus a certain rule — a kind of Jacobian of the boundary conditions with respect to the guessed variables — is used to adjust the guessed boundary conditions and control history so as to reduce the discrepancy. This process is repeated until convergence.

When used to solve problems over a long domain, shooting technique tends to be ill-conditioned due to the fact that small variations of the controls in the first part of the integration domain generally causes large variations of the final state. Multiple shooting tries to alleviate this issue by splitting the integration domain into n subintervals, and applying then the shooting technique on each interval. In multiple shooting, the unknown variables that take part in the solving process are not only the initial or final boundary conditions and the control law, but also the value of the integrated variables at the beginning and at the end of each subinterval.

Finally, in the finite discretisation approach, all the state variables and Lagrange multipliers are discretised according to a certain integration scheme, e.g. midpoint, Hermite-Simpson, Runge-Kutta, orthogonal collocation. The differential equations are then written as function of the discretised variables (i.e. they are discretised), leading to a finite-dimensional algebraic equation system which is solved with the desired algorithm, e.g. Newton-Raphson.

2.3.2 Direct methods

Differently to indirect methods, direct ones follow the "first discretise, then optimise" workflow. In direct methods, the OCP is first discretised accordingly to a certain integration scheme (Euler, Hermite-Simpson, Runge-Kutta, Orthogonal Collocation [88]), then nonlinear programming is used to find the solution of the resulting finite dimensional algebraic optimisation problem. Direct methods do not make use of the optimal control theory presented in the previous pages, rather they rely on NLP theory. In the next subsections, an introduction to NLP is first given, then numerical methods used to solve OCPs through the direct approach will be briefly described.

2.3.2.1 Introduction to nonlinear programming

Nonlinear Programming (NLP) allows to solve an optimisation problem defined by a finite set of equality or inequality algebraic constraints over a finite set of algebraic unknowns variables, together with a cost function (target) that has to be minimised. As the name suggests, both the constraints and the cost function may be nonlinear. The general problem type that NLP deals with is the following:

$$\text{find } \boldsymbol{x}^* = \arg \min_{\boldsymbol{x} \in \mathcal{R}^{n_x}} \; f(\boldsymbol{x}) \tag{2.22a}$$

subject to:

$$h_i(\boldsymbol{x}) = 0, \quad i = 1, \ldots, n_h \tag{2.22b}$$

$$g_i(\boldsymbol{x}) \leq 0, \quad i = 1, \ldots, n_g \tag{2.22c}$$

where $\boldsymbol{x} \in \mathcal{R}^{n_x}$ is the unknown vector, f is the cost function to minimise, h_i are a set of equality constraints and g_i are the inequality ones[5].

[5] Inequality constraints are sometimes omitted in the NLP formulation. Indeed, they could be taken into considerations by introducing for each inequality constraint g_i a slack variables z_i, with the constraint $z_i = g_i$, and solving then the augmented NLP problem for $z_i < 0$.

As in the case of the indirect methods, there exist some necessary conditions that the minimum of problem (2.22) must satisfy; these conditions are called the Karush-Kuhn-Tucker (KKT) conditions [10]. Before to present the KKT conditions, some definitions are first given.

Definition 4 *Active inequality set.*
The inequality constraint $g_i(\boldsymbol{x}^*) \leq 0$ is active at the point $\boldsymbol{x}^*$ if $g_i(\boldsymbol{x}^*) = 0$. Moreover, the set $A(\boldsymbol{x}^*)$ that contains the indices of all the active inequality constraints in $\boldsymbol{x}^*$ is called the active constraints set, i.e. $j \in A(\boldsymbol{x}^*)$ if $g_j(\boldsymbol{x}^*) \leq 0$ is active.

Definition 5 *Regular point.*
$\boldsymbol{x}^* \in \mathcal{R}^{n_x}$ is a regular point if the gradients of the active inequality constraints and the gradients of the equality constraints are linearly independent at $\boldsymbol{x}^*$.

Proposition 1 *Karush-Kuhn-Tucker conditions.*
Let x^* be a local minimum for the problem (2.22), and let f, h_i, g_i be continuously differentiable and let x^* be regular. Let $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu})$ be the Lagrangian function of the problem, defined as follows:

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}^\mathsf{T} \boldsymbol{h}(\boldsymbol{x}) + \boldsymbol{\mu}^\mathsf{T} \boldsymbol{g}(\boldsymbol{x}) \tag{2.23}$$

Then there exists unique Lagrange multiplier vectors $\boldsymbol{\lambda}^* \mathcal{R}^{n_h}$, $\boldsymbol{\mu}^* \mathcal{R}^{n_g}$ such that:

$$\nabla_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) = 0 \tag{2.24a}$$

$$\mu_j^* \geq 0, \quad j = 1, \ldots, n_g \tag{2.24b}$$

$$\mu_j^* = 0, \quad \forall j \notin A(\boldsymbol{x}^*) \tag{2.24c}$$

where $A(\boldsymbol{x}^*)$ is the set of active constraints in $\boldsymbol{x}^*$. If, in addition, f, h_i and g_i are twice continuously differentiable, then:

$$\boldsymbol{y}^\mathsf{T} \frac{\partial^2 \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu})}{\partial \boldsymbol{x}^2} \boldsymbol{y} \geq 0 \tag{2.25}$$

for all $\boldsymbol{y} \in \mathcal{R}^{n_x}$ such that:

$$\begin{aligned} \nabla_{\boldsymbol{x}} h_i(\boldsymbol{x}^*)^\mathsf{T} \boldsymbol{y} = 0, &\quad \forall i = 1, \ldots, n_h \\ \nabla_{\boldsymbol{x}} g_j(\boldsymbol{x}^*)^\mathsf{T} \boldsymbol{y} = 0, &\quad \forall j \in A(\boldsymbol{x}^*) \end{aligned} \tag{2.26}$$

The KKT conditions show a close relationship with the Lagrange multiplier theory for the constrained minimisation of algebraic functions. In the KKT statement the inequality constraints are divided in active and not active. If, using the Lagrange multiplier method, the active inequality constraints are considered as equality constraints while the inactive ones are disregarded, then the Lagrange multiplier sufficient conditions are equivalent to the KKT conditions. The last statement in proposition 1, contained in equations (2.25) and (2.26), provide the second order sufficient conditions for $\boldsymbol{x}^*$ to be a local minimum, not a maximum nor a saddle point. Equations (2.25) and (2.26) require the second order variation of the Lagrangian function in the tangent plane to the feasible region in $\boldsymbol{x}^*$ to be positive definite.

The KKT conditions can be further generalised so as not to require the regularity of the optimal point $\boldsymbol{x}^*$. The generalisation of the KKT are the Fritz John (FJ) necessary conditions that were first proposed in 1948 [46]. The statement of the FJ conditions is the following:

Proposition 2 *Fritz John conditions.*
Let x^* be a local minimum for the problem (2.22), and let f, h_i, g_i be continuously differentiable. Then there exists a scalar μ_0^* and multiplier vectors $\boldsymbol{\lambda}^* \in \mathcal{R}^{n_h}$, $\boldsymbol{\mu}^* \in \mathcal{R}^{n_g}$ such that:

$$\mu_0^* \nabla_x \boldsymbol{f}(\boldsymbol{x}^*) + \sum_{i=1}^{n_h} \lambda_i^* \nabla h_i(\boldsymbol{x}^*) + \sum_{i=1}^{n_g} \mu_i^* \nabla g_i(\boldsymbol{x}^*) = 0 \tag{2.27a}$$

$$\mu_j^* \geq 0, \quad j = 1, \dots, n_g \tag{2.27b}$$

$$\mu_0^*, \lambda_1^*, \dots, \lambda_{n_h}^*, \mu_1^*, \dots, \mu_{n_g}^* \quad \textit{are not all equal to zero} \tag{2.27c}$$

$$\begin{gathered} \forall\ N(\boldsymbol{x}^*)\ \textit{neighbourhood of}\ \boldsymbol{x}^*,\ \exists \hat{\boldsymbol{x}} \in N(\boldsymbol{x}^*) : \\ \left(\lambda_i^* h_i(\hat{\boldsymbol{x}}) > 0\ \forall i : \lambda_i^* \neq 0\right) \wedge \left(\mu_i^* g_i(\hat{\boldsymbol{x}}) > 0\ \forall i : \mu_i^* \neq 0\right) \end{gathered} \tag{2.27d}$$

In the case that the local minimum x^* is regular, then the FJ conditions are satisfied with $\mu_0^* = 1$ and the other multipliers are given by the KKT conditions. If the local minimum x^* is not regular then since the constraint gradients are linearly dependent the FJ conditions can be satisfied with $\mu_0^* = 0$.

NLP theory has been widely studied and plenty of algorithms to solve problem (2.22) have been developed, even for problems with non differentiable functions. While an thorough survey of the NLP solving algorithms is out of the scope of this thesis and can be found in [99, 10], the basic ideas of the most common ones are now presented. NLP algorithms include the *penalty* and *augmented Lagrangian* method, *sequential quadratic programming* (SQP) and *interior point* or *barrier* methods.

The *penalty* methods are used to solve problems as in (2.22) without inequality constraints. The problem is attacked by combining the cost function with penalty functions related to the constraints, leading to an unconstrained problem:

$$\text{find } \boldsymbol{x}^* = \arg \min_{\boldsymbol{x} \in \mathcal{R}^{n_x}} \; f(\boldsymbol{x}) + \frac{\mu}{2} \sum_{i=1}^{n_h} h_i(\boldsymbol{x})^2 \tag{2.28a}$$

where $\mu > 0$ is a positive penalty parameter. The unconstrained problem (2.28) is solved for a series of increasing values of μ, until the solution of the constrained optimization problem is obtained with sufficient accuracy. It may be also possible to find a local solution of (2.22) in a single step by (i.e. not solving the problem for several values of μ) by using an *exact* penalty function. In the *augmented Lagrangian* methods, the cost function is replaced by the so-called augmented Lagrangian function, which has the following expression (for equality-constrained problems):

$$\text{find } \boldsymbol{x}^* = \arg \min_{\boldsymbol{x} \in \mathcal{R}^{n_x}} \; \mathcal{L}_A(\boldsymbol{x}, \boldsymbol{\lambda}, \mu) = f(\boldsymbol{x}) - \sum_{i=1}^{n_h} \lambda_i h_i(\boldsymbol{x}) + \frac{\mu}{2} \sum_{i=1}^{n_h} h_i(\boldsymbol{x})^2 \tag{2.29a}$$

In these methods, $\boldsymbol{\lambda}$ represents some estimate of the optimal Lagrange multiplier vector (see Proposition 1). At each iterate both the value of $\boldsymbol{\lambda}$ and μ may be updated and the process is repeated until a sufficient approximation for the solution of (2.22) is obtained.

In the *sequential quadratic programming* methods, the problem (2.22) is approximated at each iteration by a quadratic programming subproblem whose solution defines the search direction. A basic SQP algorithm finds the search direction $\boldsymbol{d}_k$ at the k-th iteration as the solution of:

$$\text{find } \boldsymbol{d}^* = \arg \min_{\boldsymbol{d} \in \mathcal{R}^{n_x}} \; f(\boldsymbol{x}_k) + \nabla f(\boldsymbol{x}_k)^\intercal \boldsymbol{d} + \frac{1}{d} \boldsymbol{d}^\intercal \nabla_{xx} \mathcal{L}(\boldsymbol{x}_k, \boldsymbol{\lambda}_k, \boldsymbol{\mu}_k) \boldsymbol{d} \tag{2.30a}$$

subject to:

$$h_i(\boldsymbol{x}_k) + \nabla h_i(\boldsymbol{x}_k)^\intercal \boldsymbol{d} = 0, \quad i = 1, \ldots, n_h \tag{2.30b}$$

$$g_i(\boldsymbol{x}_k) + \nabla g_i(\boldsymbol{x}_k)^\intercal \boldsymbol{d} \leq 0, \quad i = 1, \ldots, n_g \tag{2.30c}$$

Where the Lagrangian function is the same as defined in Proposition 1. It can be noted that in such problem the cost function is an approximation of the change of the Lagrangian function in moving from $\boldsymbol{x}_k$ to $\boldsymbol{x}_k + \boldsymbol{d}$, and the constraints are a linearised version of the ones in (2.22). In the sequential linear-quadratic programming variant, the search direction $\boldsymbol{d}$ is computed in two stages. First, a linear program, obtained by omitting the quadratic term in the objective (2.30a) and adding a trust-region constraint to (2.30), is solved. Next, the step $\boldsymbol{d}$ is calculated by solving an equality-constrained subproblem where only the constraints that are active at the solution of the linear program are imposed (as equality constraints), while the others are neglected.

The underlying idea of the *interior point* or *barrier* methods is to use a barrier-penalty function to find a solution of the original NLP (2.22). Additional variables (called slack variables) $\boldsymbol{z}$ are introduced to the NLP, and the inequality constraints are moved to the cost function through penalties on the slack variables, resulting in the following NLP:

$$\text{find } \boldsymbol{x}^* = \arg \min_{\boldsymbol{x} \in \mathcal{R}^{n_x}} \; f(\boldsymbol{x}) - \mu \sum_{i=1}^{n_g} \log(-z_i) \tag{2.31a}$$

subject to:

$$h_i(\boldsymbol{x}) = 0, \quad i = 1, \ldots, n_h \tag{2.31b}$$

$$g_i(\boldsymbol{x}) - z_i = 0, \quad i = 1, \ldots, n_g \tag{2.31c}$$

where $\mu > 0$ is a positive parameter. The NLP problem is thus reduced to contain only equality constraints. The *interior point* method tries to find a solution for problem (2.31) in the limit $\mu \to 0^+$. It can be noted that at the solution $\boldsymbol{x}^*$, the slack variables z_i are equal to the inequality constraints $g_i(\boldsymbol{x}^*)$, moreover, since the target function tends to infinitum as $z_i \to 0^-$, the value of the slack variable at the solution z_i^* is negative (thus the inequality constraints are satisfied). The solution of the modified problem (2.31) is an approximation of the one of the original problem (2.22), and the approximation gets close to the "exact" solution as $\mu \to 0^+$.

The *penalty*, *augmented Lagrangian* and *interior point* approaches present many similarities with the penalty method discussed in section 2.3.1.3; in all cases, the inequality-constrained problem is reduced to an equality-constrained one by adding some penalty functions to the cost function.

2.3.2.2 Notes on numerical methods

In direct methods, the optimal control problem is translated into a finite dimensional constrained minimisation problem (2.22) which is solved using available NLP solvers. Direct methods however can differ for the variables that are discretised, i.e. both states and controls or only the controls can be discretised. The shooting and multiple shooting methods for the direct approach are very similar to the same methods for the indirect approach as described in section (2.3.1.4). In the shooting method, the controls are discretised and, together with the initial and/or final state variables, they become NLP optimisation variables. For any given control law, the differential equations are easily integrated allowing to evaluate the OCP target and constraints. The difficult task in the shooting method is the evaluation of the target variations with respect to control variations. Differently to the shooting method for the OCP indirect approach, OCP Lagrange multipliers are not considered.

As described in (2.3.1.4), the multiple shooting divides the integration domain into n intervals, and single-shooting is applied on each subintervals. The values of the state variables on the $N-1$ interface points are added to the NLP optimisation variables.

In the direct full collocation method, both the states and controls are discretised, leading to a larger NLP. The OCP differential equations and constraints are expressed as functions of the discrete variables and included as constraints in the NLP problem.

2.3.3 Considerations on indirect and direct methods

In has been shown that indirect methods allows to find the solution of optimal control problems by solving the necessary conditions for optimality. Since the necessary conditions come in the form of differential and algebraic equations together with a minimisation subproblem, in some cases a closed-form solution exists, i.e. the equations can be analytically solved. This allows indirect methods to find the exact solution for a (not so small) family of problems; differently, direct methods are used only for numerical solutions. Even if most of the OCP arising from practical problems are highly nonlinear and the necessary condition equations cannot be explicitly solved, there exists a relatively wide class of problems for which the closed-form solution exists. This class consists in all the physical systems for which the Euler-Lagrange equations of motions can be analytically solved. Indeed, the Euler-Lagrange equations can be considered as the solution of a proper Optimal control problem. Accordingly to the *Principle of Least (or Stationary) Action* [52], a physical system moves (or evolves) from an initial state $\boldsymbol{x}_i$ to a final state $\boldsymbol{x}_f$ in such a way that its *Action A* is stationary. The *Action A* is defined as the integral of the *Lagrangian* function over the time:

$$A = \int_{t_i}^{t_f} \mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}}, t) dt \tag{2.32}$$

where $\boldsymbol{q}$ are the generalised coordinates. It is easy to verify that the necessary conditions (2.10) for an unconstrained OCP with a pure Lagrange target equal to the *Action A* becomes exactly the Euler-Lagrange equations:

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{\boldsymbol{q}}}\right) - \frac{\partial \mathcal{L}}{\partial \boldsymbol{q}} = \boldsymbol{0} \tag{2.33}$$

This establishes a fascinating link between optimal control theory and the theory of Lagrangian mechanics.

A second advantage of indirect methods over direct ones from a theoretical point of view is the calculus of the Lagrange multipliers. The Lagrange multipliers, despite they may seem additional variables merely introduced to find the optimality conditions, have a relevant physical meaning. Indeed, the Lagrange multiplier associated to a certain constraint expresses the rate of change of the target with respect to the violation of that constraint. Therefore the Lagrange multipliers are very useful in sensitivity analyses, where the variation of the target with respect to variation of the problem data is studied. Actually, it is not completely correct to state that direct methods do not allow to calculate the OCP Lagrange multipliers. It has been shown that the KKT conditions (1) make use of (NLP) Lagrange multipliers, which are clearly different from the OCP ones, but a close relationship between them exists. Such relationship is not general but strictly depends on the discretisation scheme used to transcribe the OCP into the NLP problem. In chapter 6 this relationship will be derived for the midpoint discretisation scheme.

Chapter 3

Indirect and direct methods for optimal control minimum lap time simulations

Minimum lap time simulations are nowadays a tool widely used to optimise the design and setup of road vehicles. Even if several approaches can be adopted to face such problems, optimal control is the strategy that, to the best knowledge of the author, has been used in the most relevant works in literature. Both indirect and direct methods have been intensively used for minimum lap time simulations, yet it is still not clear whether one of the two approaches is more suitable (e.g. fast, robust, effective) than the other. In this chapter direct and indirect methods are compared on three three vehicle optimal control test problems in order to understand what are the differences between the two methods and if one is more effective in solving minimum lap time OCPs.

3.1 Introduction

In the last decades the applications developed for minimum lap time problems have become a tool widely used to improve the performance of race vehicles. Minimum lap time problems not only are of great practical interest to help design, optimise and setup a vehicle for maximum performance but they also are a challenging theoretical and numerical problems. Early attempts to solve minimum lap time problems date back to the late 50's [82]. Later in the 80's authors of [68] simulated a section of the Formula One Circuit Paul Ricard in southern France using a quasi-steady state optimization routine to compute the optimal controls. Since then, many improvements have been introduced for solving minimum lap time problems with several authors proposing and developing different solving techniques and new theoretical and numerically efficient algorithms also supported by the increase of the cpu performance. Among all the methods that have been used until now, the ones that showed the best capabilities falls into four major categories: quasi-steady state, optimal control based, driver model based and evolutionary algorithm based simulations.

In the quasi-steady state (QSS) approach, the racing line is provided as input and it is divided into segments in which the vehicle is considered to be in stationary conditions except for few state variables such as the speed along the racing line. The curvature radius is then calculated for each segment and the vehicle is assumed to have zero longitudinal acceleration in those points where the curvature has a local maximum (at the corner apex); thus the maximum vehicle speed can be calculated in correspondence of such points. Then, the vehicle speed is reconstructed backward and forward on the basis of the maximum g-g envelope. Examples of this method can be found in [36, 84, 17, 18, 19, 81]. This method has shown both good robustness and fast computation times, together with the capability to use complex vehicle multibody models; however most of the transient behaviours is neglected (e.g. tyre loads dynamics, yaw dynamics and suspension damper effects) and the results obtained are less accurate when compared with other minimum time techniques. An extended iterative steady-state approach was presented [94] to include some of these effects, e.g. the suspension damping effects. Despite the additional benefits compared to standard QSS method a comparison with optimal transient solution is not given.

The second family of minimum lap time methods falls into the optimal control problem (OCP) theory, which are also called transient-optimal control to distinguish them from quasi steady state simulations [21, 17]. The general idea behind this approach is to translate the minimum lap time problem into an optimal control problem, where the dynamic equations of motion become constraints of the optimisation process. As described in chapter 2, methods for solving optimal control problems can be mainly grouped into four main categories: dynamic programming (DP), indirect optimal control, direct optimal control and evolutionary algorithms (EA).

To the best of the author's knowledge, despite the theoretical advantages of the Dynamic Programming, such as handling discrete/continuous variables and guarantee the global optimum, none minimum lap time application has been solved with this method in the literature. The main reason is that, as described in chapter 2, it suffers from the curse of dimensionality even with relatively small vehicle models [86, 12]. Even if hybrid differential dynamic programming seems a promising method, it is still in the early development stage and it has not been applied to challenging minimum lap time problems yet.

Indirect and direct methods for optimal control problems, already described respectively in sections 2.3.1 and 2.3.2, have been widely used for minimum lap time simulations since mid 90's. Early examples of application of indirect optimal control theory this kind of simulations are reported in [44, 30]. Since the late 90's various other works have used this approach both for motorbike lap time simulations [8, 85] and car lap time simulations [90, 91, 14, 59].

Among the direct methods, direct multiple shooting method (also known as parallel shooting method) emerged as the one of the most efficient because it is less affected by high sensitivities and naturally renders itself for parallelisation [22]. In [50] it was successfully applied to minimum time problem with gear choice using using *MUSCOD-II* software [33] and with a partial outer convexification to handle discrete variables. Methods based on direct full collocation are largely diffused in the community because the resulting NLP can be easily solved thanks to the availability of *IPOPT* [95], a robust and efficient interior point algorithm. The approaches mainly differ from the discretisation method but *IPOPT* is the solver used by all of them, e.g. [7] used Lagrange Polynomials to discretise both control and states, [72] used a trapezoidal integration scheme to convert the optimal control problem into a NLP problem by means of ICLOCS toolbox [35], various minimum lap time problems were formulated with a direct orthogonal collocation method based on Radau pseudo-spectral scheme by means of the software package *GPOPS-II* [71], which proved to be effective to illustrate the impact of optimal usage of energy recovery systems on fuel consumption saving [58] and the impact of aero-suspension interactions and adjustments on the lap-time performance of the car [65]. Direct methods have rarely been applied to motorbike models; to the author's knowledge, the only relevant work is [83] where a full lap was simulated with a relatively complex motorbike model. However, he did not optimised at once the trajectory and the controls but a path following algorithm was used to make the motorbike follow a given trajectory.

Many authors proposed various alternative techniques in order to increase the robustness and to reduce the computational burden due to the size of the NLP resulting from long circuit/test course. One of these are the moving horizon techniques, which decompose the global optimal control problem into a sequence of local optimal control problems over a finite horizon (i.e. preview length) that is moving forward in time [37, 40, 48, 49] and satisfies appropriate continuity conditions. Optimisation of the preview length is necessary to guarantee for finding the global problem optimum. Additionally, this technique cannot be directly used when optimisation of global variables is required. Another proposed idea splits the problem into trajectory *planning* and *tracking*. Trajectory planning is usually solved as an optimisation problem on a simplified vehicle model and the tracking task is performed with a *driver model* (or controller) that guides the full vehicle along the pre-calculated trajectory. Examples are [16] that combines geometrical trajectory and speed profile optimisation with a simple driver model; [24], and [47] used Model Predictive Control to implement the tracking task. A noticeable example applied to motorcycle is [83] where a full lap was simulated with a relatively complex motorbike model using a MPC path following algorithm to track a given trajectory. However, all these approaches, despite their ability to be used in combination with complex vehicle models, should be considered as sub-optimal solutions since the driver model influences the lap time obtained.

Finally, evolutionary approaches, such as genetic algorithms, are the alternative numerical algorithms, for the solution of direct sequential optimal control, compared to derivative based NLP solvers. A recent application to minimum lap time problem is reported in [63, 62]. The procedure that leads to the lap time minimisation is the following: for a given control history defined as piecewise linear, the equations of motion are integrated, resulting in a certain manoeuvring time; then the genetic algorithm optimises the control values at given number of knots in unknown time positions and the equations of motions are integrated with the new controls to evaluate the cost and the constraints. The genetic algorithm thus selects the control variations that brought to an improvement of the performance and the process is iterated again in a manner very similar to a single shooting methods. This approach demonstrated to be able to handle very complex vehicle models, however it also resulted to be significantly slow: the computation of a manoeuvre over two turns takes approximately one day to execute and the control law shape is derived from experimental data.

As final a remark of the literature review above, authors think it is worth it investigating benefits and limitations of optimal control methods for minimum lap time problems that exhibit the following characteristics:

- they possibility cope with non trivial vehicle models,e.g. several dof ($> 5-7$) including highly non-linear tyre model and aerodynamics interaction;
- they can provide simultaneous optimisation of the racing line and the controls;
- they have the ability to compute a simulation in a reasonable amount of time (e.g. less than few hours);
- they are sufficiently robust to variations on initial guess.

According to the literature review and to the authors knowledge and experience, three approaches emerged as the most effective to solve minimum lap time problems.

Following the historical development, the first method is an indirect approach implemented by the software *Pins* (formerly known as *XOptima*), which is described in [8]. It has been used since the late 90's and the most representative minimum lap time results achieved with *Pins* are presented in [8, 85, 27] for motorbike applications, and in [90, 91] for cars. The second method is a direct multiple shooting method implemented using the software *MUSCOD-II* and whose most successful result is in [50]. The last and most recent method is based on direct optimal control with full direct transcription via pseudo-spectral collocation and it is represented by the software *GPOPS-II*, [71]. It was released quite recently in the 2013 and its state of the art applications in this topic is described in [58, 55].

To the best author's knowledge, in the literature, there are no comparisons of the accuracy of the solution and numerical performance on minimum lap time problems among these methods. This gave the motivation for this study to compare the most promising optimal control approaches on the same optimal control problems, in order to highlight the similarities and the differences, and possibly understand which one is more indicated for a particular purpose. Unfortunately *MUSCOD-II* is not available and for this reason the chapter only focuses on the comparison of *Pins* and *GPOPS-II* software as representative of state of art methods of indirect and direct approach respectively.

In section 3.2, *Pins* and *GPOPS-II* and their approach are briefly presented, then in section 3.3 they are tested against three vehicle optimal control problems. The comparison will focus on the solution accuracy, robustness with respect to guess perturbations, constraints enforcement and parameter sensitivity. Finally in section 3.4, the results are summarised and the software performance discussed.

3.2 Minimum time manoeuvring OCP

In general, a minimum lap time problem can be formulated as an optimal control problem (see (2.6)) where the target function equals the lap time, and the first order differential equations are the equations of motion of the vehicle. Constraints are added to ensure the simulation withstand physical limits like track boundaries, power used, tyre adherence.

The optimal control problem defined in (2.6) is quite general and may not fit the implementation of various software for solving optimal control problems. This is also the case for the software (*Pins*, *GPOPS-II*) that we have selected for comparison. In these cases some problem reformulations of the OCP problem may be required.Thus next subsections 3.2.1, 3.2.2 briefly describe the two software with key features and limitations.

3.2.1 OCP: formulation and solution with PINS

Pins is a collection of libraries and programs mainly developed to symbolically formulate and numerically solve Optimal Control Problems (OCPs) for non linear dynamical systems described by differential equations. *Pins* implements an indirect method with penalties and barriers to handle generic mixed state and control constraints. The optimal controls are explicitly derived using the Pontryagin Maximum Principle, which is done formally building a map that computes the controls as a function of state and co-state. The map is solved analytically whenever the problem makes it possible, otherwise a numerical procedure is used to obtain controls and their derivatives with respect to the states and co-states. The resulting Two-Points-Boundary-Value-Problem (TPBVP) is approximated with finite difference (based on midpoint quadrature) to give a non-linear algebraic system of equations, which is solved with a dumped Newton Affine Algorithm specifically developed to exploit the Bordered Almost Block Diagonal (BABD) structure of the Jacobian [2]. *Pins* makes uses of the *Maple©* symbolic engine, via the *XOptima* package, to symbolically formulate the optimal control problem, automatically generate the equations of necessary conditions of optimality and the corresponding analytical Jacobians, and finally translate into *c++* code. The generated code can be, if necessary, further manipulated by the user and compiled and linked with libraries contained in *MechatronixToolKit* by the program *pins*, in order to produce a stand alone program or a callable library that can be used in custom code for different purposes (including real time applications). The *MechatronixToolKit* is a collection of *c++* libraries of classes such as Non Linear system solvers, Boundary Value Problem Solvers, ODE-DAE solvers. The library is also complemented with utility classes such as vehicle components (e.g. tyre models, internal combustion engine models, etc), 2D-3D road models, splines and various interfaces to other languages such as *Mruby* and *Lua* and *MATLAB©*/Simulink.

The OCP formulation in *Pins* allows for differential equations of the form:

$$\boldsymbol{A}(\boldsymbol{x}, \boldsymbol{p}, \zeta)\dot{\boldsymbol{x}} = \tilde{\boldsymbol{f}}\,(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{p}, \zeta) \tag{3.1}$$

where $\boldsymbol{A}$ is a square matrix that cannot depend on the controls. Multibody and vehicle dynamic equations can always be written in this form. In practice, the main difference between 2.6c and (3.1) is that the latter allows the differential equations to be implicit provided they are linear in the velocities. This is an advantage compared to the majority of OCP solvers that require the differential equations to be in the explicit form 2.6c where inversion of matrix $\boldsymbol{A}$ may be necessary as also found in almost all text books [76, 20, 75, 11].

Pins does not directly handle integral constraints (2.6f) but they can be added to the problem by converting the integral constraints into differential constraints introducing additional states with proper boundary conditions.

Pins treats inequalities (2.6e) augmenting the target function (2.6a) with a weighted sum of penalty or barrier functions $p(\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{p}, \zeta))$ for each inequality. The function p is designed to be continuously differentiable of class C^3 in such a way that it evaluates nearly zero (i.e. ϵ) when $\boldsymbol{g}$ is at a distance h from bound. It then grows to 1 at the bound then pretty linearly to infinity out of the bound for the penalty and to infinite at the bound for barrier. Clearly barrier does not allow to break the bound. Parameters h and ϵ are used respectively to define when the penalty starts to increase the cost and how much.

The main limitation of *Pins* is the fact that cannot directly handle DAEs of index greater equal 1. For what concerns index-1 algebraic constrains only those linear in the algebraic variables can be used. For DAEs with index of higher order reduction techniques and penalisation terms in the cost function (2.6a) to avoid constraint drift are necessary [38].

Since *Pins* implements an indirect method it has to solve the necessary conditions of optimality which are described by a pure TPBVP since penalty are also used to enforce constraints on controls. The TPBVP consists of a set of n_x differential equations for the states $\boldsymbol{x}$, n_x adjoint equations for the Lagrange multipliers $\boldsymbol{\lambda}$, n_u algebraic equations for the optimal controls $\boldsymbol{u}$ and $2n_x + n_p$ equations for the boundary conditions. The controls are formally solved either symbolically or numerically as a function of the states $\boldsymbol{x}$ and Lagrange multipliers $\boldsymbol{\lambda}$. The TPBVP is discretised using a finite difference midpoint scheme and the resulting large set of algebraic equations is solved with a dumped Newton Affine scheme that exploits the Jacobian block diagonal structure [8],[9].

Summarising, the key points of *Pins* approach for the solution of the indirect optimal control problem are:

- dynamic equations (2.6c) can be given in implicit form, but it has to be linear in the state derivatives;
- inequality constraints (2.6e) are treated with penalty/barrier functions (constrained problem converted in equivalent unconstrained problem)
- only index-1 algebraic constraints linear in the algebraic variable can be handled directly
- the controls are formally solved analytically, if an explicit analytic solution is not available numerical methods are used
- states are discretised on the mesh points while controls are assumed constants on cells and thus the TP-BVP problem is solved as a large set of algebraic equations roughly of dimension $(2\,n_x + n_u)\,N + 2\,n_x + n_p$ where N is the number of mesh points.
- solution is obtained using custom nonlinear system numerical solver

Further details can be found in [8],[9].

3.2.2 OCP: formulation and solution with GPOPS

GPOPS-II is a *MATLAB©* software intended to solve general optimal control problems for non linear dynamical systems described by differential-algebraic equations. *GPOPS-II* implements a direct full collocation approach by means of pseudo spectral method. The continuous-time optimal control problem is approximated using a new class of a variable-order Legendre-Gauss-Radau quadrature orthogonal collocation polynomials resulting into a sparse nonlinear programming problem. This NLP is then solved using either the NLP solver *IPOPT* or the NLP solver *SNOPT*. A distinguishing feature is the adaptive mesh refinement method that determines the number of mesh intervals and the degree of the approximating polynomial within each mesh interval to achieve a specified accuracy. To achieve this *GPOPS-II* performs an a-posteriori solution error estimation and refine the mesh in those mesh segments where the error is higher than a certain threshold; this process is then repeated until all mesh interval satisfy the desired error threshold. Adopting the variable order for each mesh interval and the mesh refinement *GPOPS-II* is able to achieve high accuracy limiting the use of resources by putting refining only where it is necessary. By means of the free *MATLAB©* package ADIGATOR [98] symbolic gradients and Jacobians could also be generated.

As for the general formulation of the OCP described by (2.6), *GPOPS-II* fits the definition, thus it requires the differential equations to be explicit. In general, vehicle dynamic equations can often be reduced to the explicit form. However, it may not be possible when the model complexity increase, which is the case of advanced motorcycle models, e.g. [85, 27, 28]). For some problems the implicit form is more robust and the solver converge faster to solution [31]. Since *GPOPS-II* naturally treats algebraic equations of index 1 the implicit formulation can be also implemented at the cost of doubling the problem dimension adding n additional states $\boldsymbol{y}$ and n additional algebraic path constraints as follows

$$\dot{\boldsymbol{x}} = \boldsymbol{y} \tag{3.2}$$

$$\boldsymbol{0} = \tilde{\boldsymbol{f}}\,(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\beta}, t) - \boldsymbol{A}(\boldsymbol{x}, \boldsymbol{\beta}, t)\boldsymbol{y} \tag{3.3}$$

GPOPS-II approximate both states and controls using multiple-interval Legendre-Gauss-Radau quadrature orthogonal collocation method that transcribes the OCP into a NLP of dimension roughly of $n_x(N^{(k)} + 1) + n_u N^{(k)}$ for each $k = 1 \dots K$ intervals. The structure of the Jacobians and Hessian are sparse matrices that do not show any specific structure that can be exploited by the NLP solver. Additionally, since *GPOPS-II* uses *IPOPT* as NL solver internally the problem is augmented to handle the equality constraints via Lagrange multipliers. This is transparent from the user perspective but has an effect on the computational performance point of view. Additionally, *IPOPT*, being an interior point method, uses barrier functions to treat inequality constraints [95]. However, it adopts several automatic strategies to adapts the barrier weights during algorithm convergence to satisfy the inequalities with the higher accuracy possible .

Summarising, the key points of *GPOPS-II* in the direct solution of the OCP are:

- dynamic equations (2.6c) must be in explicit form otherwise the dynamic system dimension has to be doubled;
- algebraic constraints index-1 can be used
- automatic mesh refinement algorithm available
- automatic differentiation is available via Adigator to generate the necessary gradients, Hessian and Jacobians
- the solution is obtained using NLP solver *IPOPT* that implements a robust and quite fast interior point algorithm.

Further details on the software can be found in [71].

3.3 Test bench examples for vehicle optimal control problems

In this section the two software previously presented, *Pins* and *GPOPS-II*, are tested on three test bench vehicle optimal control problems. The first case of study deals with a simple motorcycle model; it has been chosen because the exact solution can be mathematically derived, thus it can be used for comparison with numerically calculated solutions. The second test benchmark consists in the reconstruction of a race circuit from experimental data; since this OCP does not include any path constraints, it leads to the same minimisation problem both through the indirect and direct approach. For this reason this problem has been chosen to test the robustness of *Pins* and *GPOPS-II* with respect to perturbations in the guess. Finally, the third and last test problem consists in a minimum lap time problem of a relatively simple, yet effective, car model; this represents a typical utilisation scenario for the studied software. The constraint enforcement, sensitivity to parameter variations and robustness to guess perturbations will be studied.

3.3.1 Basic two-wheeled minimum manoeuvre

The first optimal control problem consists in the minimum manoeuvre time of a basic motorcycle model, that has to be moved from the upright configuration (zero roll angle) to a leaned configuration (non-zero one) in the minimum time. The assumptions are that the tyres have zero slip, the suspensions are fixed and the speed is constant. Under these circumstances the model has one degree of freedom only, the roll angle, see Ch. 6 of [89].

The motorcycle state space model is described by two state variables, the roll angle ϕ and the roll rate ϕ_{dot}, and one input, the steering angle δ. The related first order differential equations are

$$\dot{\phi} = \phi_{dot}, \qquad I_{xx}\dot{\phi}_{dot} = mh\left(g\phi - \frac{V^2}{L}\delta\right) \tag{3.4}$$

where m is the total (vehicle plus rider) mass, I_{xx} the roll moment of inertia about line joining the two tyre contact points, h the distance height of the centre of mass from ground, L is the wheelbase and g is the gravity. Equation (3.4) can be conveniently rewritten in order to highlight the independent model parameters:

$$\dot{\phi}_{dot} = \frac{mgh}{I_{xx}}\left(\phi - \frac{V^2}{gL}\delta\right) \equiv A\left(\phi - B\delta\right) \qquad \text{where} \qquad A = \frac{mgh}{I_{xx}} \qquad B = \frac{V^2}{gL} \tag{3.5}$$

The minimum time to roll problem can be formulated as follows:

$$\begin{aligned} \underset{t_f \geq t_0}{\text{minimize}} \quad & t_f \\ \text{subject to ODEs:} \quad & \dot{\phi} = \phi_{dot}, \qquad \dot{\phi}_{dot} = A\left(\phi - B\delta\right) \\ \text{and constraints:} \quad & |\delta| \leq \delta^{\max}, \quad \phi(t_0) = 0, \quad \phi(t_f) = \phi_f, \\ & \qquad \phi_{dot}(t_0) = 0, \quad \phi(t_f)_{dot} = 0 \end{aligned} \tag{3.6}$$

The analytical solution is obtained using the PMP [89]:

$$t_f = \frac{1}{\sqrt{A}}\big(\ln(w) - \ln(1+f)\big), \qquad \delta = \operatorname{sign}\phi_f\, \delta^{\max} \times \begin{cases} +1 & 0 \leq t < t^* \\ -1 & t^* \leq t \leq t_f \end{cases} \tag{3.7}$$

where

$$f = -\frac{|\phi_f|}{B\delta^{\max}}, \quad w = 1 - f - \frac{f^2}{2} + \frac{\sqrt{f(f+4)(f^2-4)}}{2}, \quad t^* = \frac{\ln(1+w) - \ln(2)}{\ln(w) - \ln(1+f)} \tag{3.8}$$

Finally, the solution exists only if $f > -1$ and w is real $w > -1$ which imply $f \in (-1, 0]$ and:

$$|\phi_f| < \delta^{\max} B \tag{3.9}$$

It is worth noting that a positive steer angle δ corresponds to a positive roll angle ϕ in steady state conditions, while the bike is performing a right or clockwise turn.

Solution analysis

The numerical solution obtained using the motorcycle dataset of table 3.1 ($A = 42.92s^{-1}$ and $B = 8.551$). The theoretical minimum time with the data used here is $t_f = 1.07236889954 \times 10^{-1}s$.

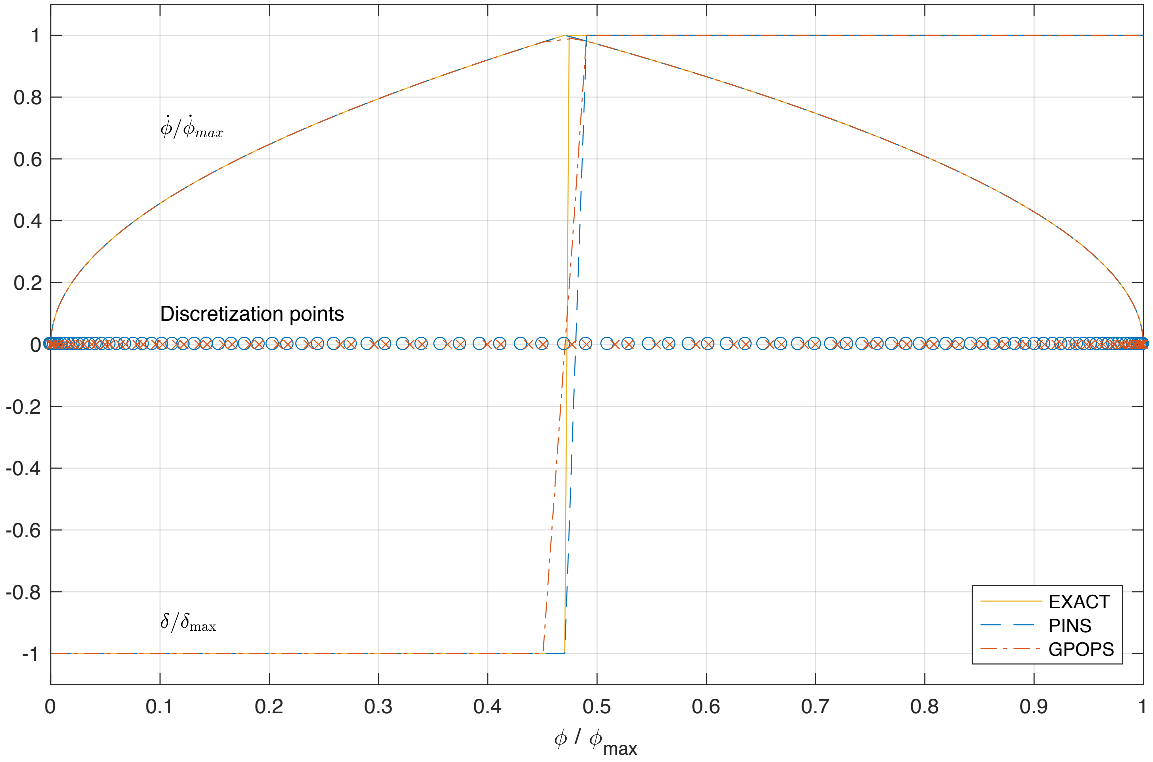


Figure 3.1: The control δ and roll rate $\dot{\phi}$ are shown as function of the roll angle ϕ. The solutions provided by *Pins* and *GPOPS-II* are shown together with the exact one. Blue circles represent *Pins* discretisation points, red crosses *GPOPS-II* ones.

The solution given by the two solvers on a mesh with 100 discretisation points is shown in figure 3.1. Settings used for *GPOPS-II* are reported in table 3.2. It is noted that *Pins* uses an equally spaced grid while *GPOPS-II* puts the discretisation points at the Legendre-Gauss-Radau points [1], according to to the LGR method therein implemented. Figure 3.1 highlights the bang-bang behaviour of the control δ, which switches from $-\delta_{\max}$ (steer opposite to the direction of turning) to $+\delta_{\max}$ (steer in the direction of turning) at a time equal to $t^* \approx 0.47t_f \approx 5 \times 10^{-2}s$. Both solvers require one mesh interval to capture the complete change in the control; for this reason, *GPOPS-II* takes three discretisation points

Table 3.1: Motorcycle problem dataset

variable	value	description
g	$9.806\,[m/s^2]$	gravity
V	$11\,[m/s]$	speed
m	$273\,[kg]$	total mass
L	$1.443\,[m]$	wheelbase
I_x	$70.79\,[kg\,m^2]$	roll inertia moment
$\delta^{\max}$	20°	max steer angle
ϕ_f	20°	final roll angle
t_f^g	$1\,[s]$	guess on final time
ϕ^g	$\phi_f t/t_f^g$	guess on roll angle

to capture the control switch[1], while *Pins* takes only two. The roll rate in figure 3.1 shows an apex in correspondence of t^*, as consequence of the change in the control.

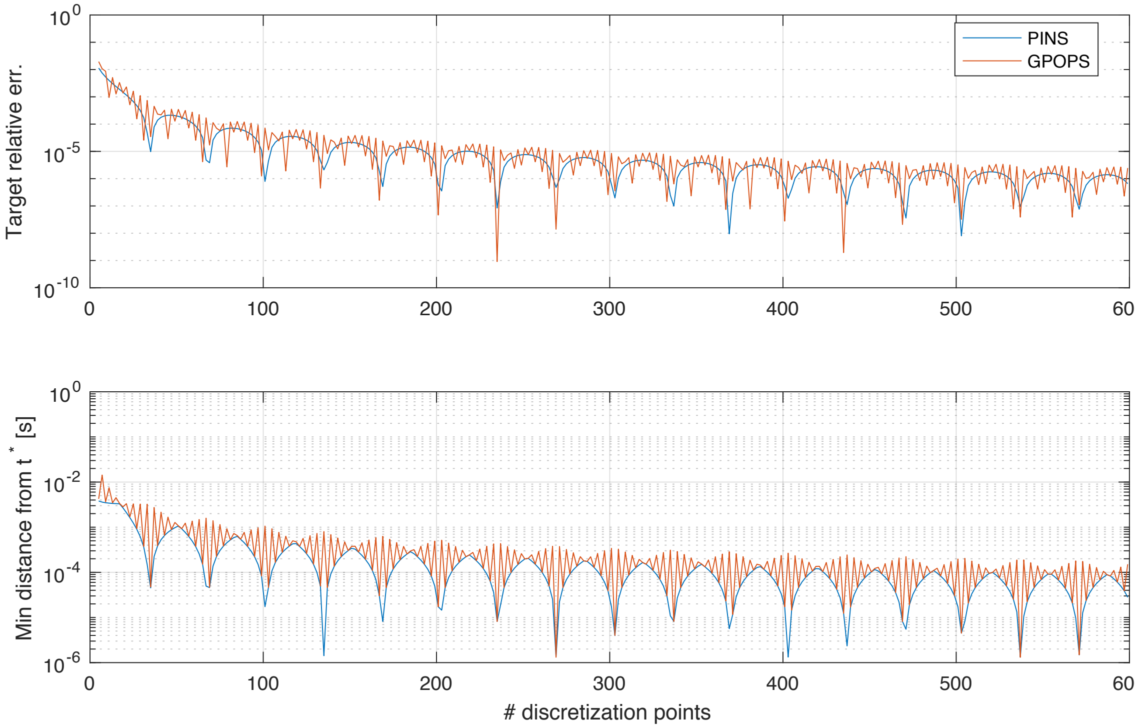


Figure 3.2: The relative error to the exact final time t_f (top plot) and the distance d^* of the time switching instant t^* from the nearest discretisation points (bottom plot) are shown as function of the number of discretisation points. The two plots suggest that the solution accuracy is dominated by the capability to capture the bang-bang trend in the control δ.

[1] Collocation points captures accurately smooth functions, but discontinuities in non-smooth solutions can be captured only at mesh interval boundaries.

The accuracy of the target value t_f provided by the two solvers is shown in figure 3.2 as a function of the number of discretisation points used. The error presents an wavy trend, which, at a first glance, may appear quite strange, as one would expect that the higher is the number of discretisation points the more accurate is the solution. Taking into consideration that the control δ has a bang-bang behaviour, the accuracy of the target value t_f strictly depends on the ability of the integration scheme to capture the discontinuity of the solution. In particular, the capability of the integration scheme to capture the control discontinuity depends on the distance d^* of the exact switching time t^* from the nearest mesh point: the higher d^* is, the less accurately the control discontinuity is captured. The distance d^* is shown as function of the number of discretisation points in the bottom plot of figure 3.2. It is evident that the wavy trend of the target error is exactly the same of the distance d^*, thus the accuracy of the solution is dominated by the capability to capture the discontinuity of the δ control at t^*.

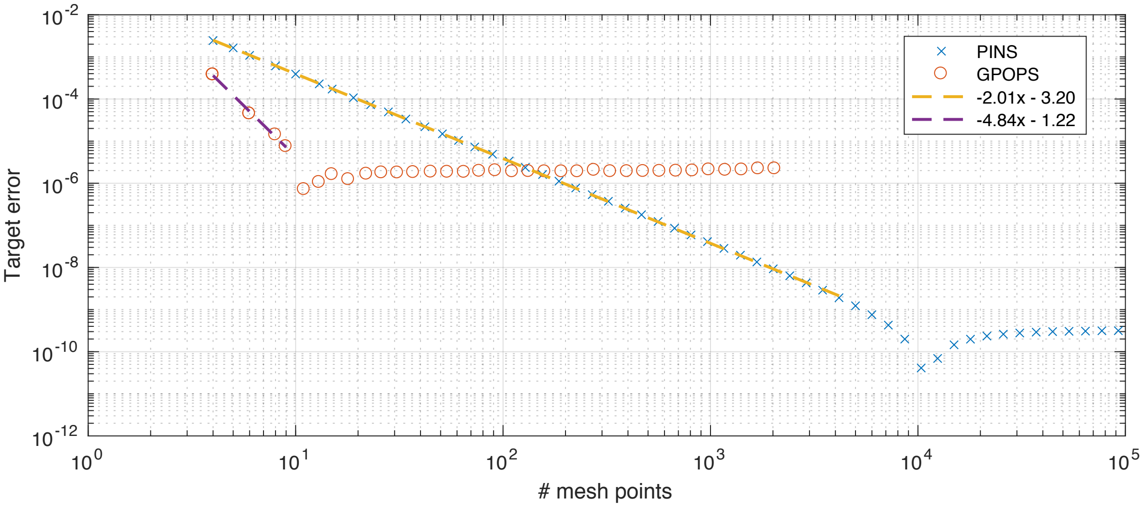


Figure 3.3: The relative error to the exact final time t_f is shown as function of the number of discretisation points used in the intervals $t \leq 0.46t_f$ and $t \geq 0.48t_f$. In the interval $0.46t_f \leq t \leq 0.48t_f$, 10^4 fixed discretisation points have been used. It can be noted that the straight lines fitting the error trends have a slope which is in agreement with the integration accuracy order of the two solvers.

In order to better analyse the accuracy of the solution, the previous analysis has been re-performed using a large and fixed number of discretisation points near the switching time t^*. More precisely, 10^4 fixed discretisation points has been used in the interval $0.46t_f \leq t \leq 0.48t_f$, while only the number of discretisation points N in the rest of the domain has been changed. The results obtained are presented in figure 3.3: *GPOPS-II* shows a steep decrease of the target error, and even with few points ≈ 10 it reaches a relative error of only

10^{-6}. The fitting highlights an accuracy, which goes as $\approx N^{-4.8}$. For a number of mesh points approximately greater than ten, the solution error does not decrease further and remains stuck at $\approx 10^{-6}$ even at $N = 10^3$. *Pins* differently presents a less rapid decrease of the solution error $\approx N^{-2}$, but increasing the number of mesh points it reaches a lower error; with $N \approx 10^4$ its error is $\approx 10^{-10}$. Numeric noise does not allow *Pins* to go below such error.

The authors increased the number of mesh point in the interval $0.46t_f \leq t \leq 0.48t_f$ (up to 10^5) and even tuned *GPOPS-II* parameters (e.g. using auto mesh refinement and/or lowering *IPOPT* tolerances, and/or imposing a mesh point at the switching point $t = t^*$) in order to achieve a better accuracy for the *GPOPS-II* solution. However, the best accuracy obtained was always not lower than $\approx 10^{-7}$. Moreover, when trying to increase the solution accuracy in *GPOPS-II*using a very fine mesh (either with fixed points or using the automatic mesh refinement), the solution so obtained suffered of control oscillations near the switching point $t = t^*$, as shown in figure 3.4. It is opinion of the authors that if the controls could be analytically solved (as it is done in *Pins*) these oscillations may be reduced and the accuracy of the solution increased. However this cannot be done in *GPOPS-II* since the Lagrange multipliers are not available when the evaluation of the first order equations is performed.

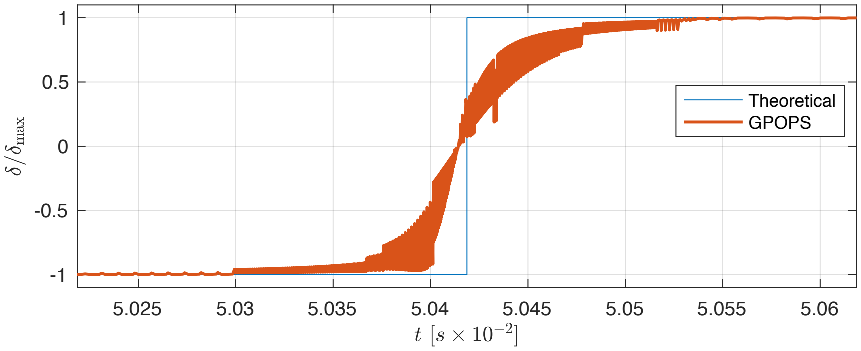


Figure 3.4: Detail of *GPOPS-II* solution control near the switching point t^*. The solution is calculated on a fine mesh (mesh tolerance error $\leq 10^{-11}$. The control is affected by noticeable oscillations that prevent the solution final time accuracy to be less then 10^{-7}.

This case of study shows that *GPOPS-II* uses a better integration scheme that allows to achieve a good accuracy (relative to *Pins*) when the solution is smooth and few discretisation points are used. On the other side *Pins* demonstrates to be more accurate with a relative large number of discretisation points and to be able to get closer to the exact solution.

Table 3.2: *GPOPS-II* settings

variable	value
minimum collocation points	2
maximum collocation points	2
IPOPT tolerance	10^{-8}
method	RPM-Differentiation
scaling	automatic-bounds
linear solver	ma57 [34]

3.3.2 Track reconstruction

The second OCP test consists in the reconstruction of a circuit from experimental data. In lap time simulations it is common to use a curvilinear abscissa approach [25, 59, 57] to track the vehicle position along the circuit. Thus the knowledge of the track geometry is fundamental for such purpose.

The road can be modelled as described in appendix A, where the geometry is given as the road reference frame $\boldsymbol{W}_r$ and the road width r_w as a function of the centre-line curvilinear abscissa s. As described in appendix A, the road can be described by means of seven variables: the three coordinates of the road centre-line position $x_r(s)$, $y_r(s)$, $z_r(s)$, together with the three angles determining the orientation matrix $\psi(s)$, $\sigma(s)$, $\beta(s)$ and one additional variable for the road width $r_w(s)$. An effective approach to reconstruct the road geometry — i.e. these seven variables — from a 3D map of the circuit is presented in appendix B. This corresponds to an optimal control problem (B.3), where the target to minimise is the error between the road borders and the reference borders obtained from the 3D map. The states of the OCP are the variables determining the road position, orientation and width, the controls are the rate of change of the road orientation and width. The optimal control problem (B.3) is in the form of an constrained least-square problem, where the constraints are originated by the first-order equations No path (state) constraints are added to the OCP formulation (B.3), therefore the actual OCP solved by *Pins* and *GPOPS-II* is exactly the same (even if on slightly different meshes).

Solution analysis

Pins and *GPOPS-II* have been tested on the reconstruction problem of four different circuits, two three-dimensional and two two-dimensional: Adria (Italy, 2D), Montmelò (Spain, 2D), Imola (Italy, 3D) and Mugello (Italy, 3D). The numeric dataset used to feed the OCP (B.3), in particular the target expressions (B.6b), are reported in table 3.3; *GPOPS-II* settings are the same of the previous example (see table 3.2). A mesh grid size of 1 meter

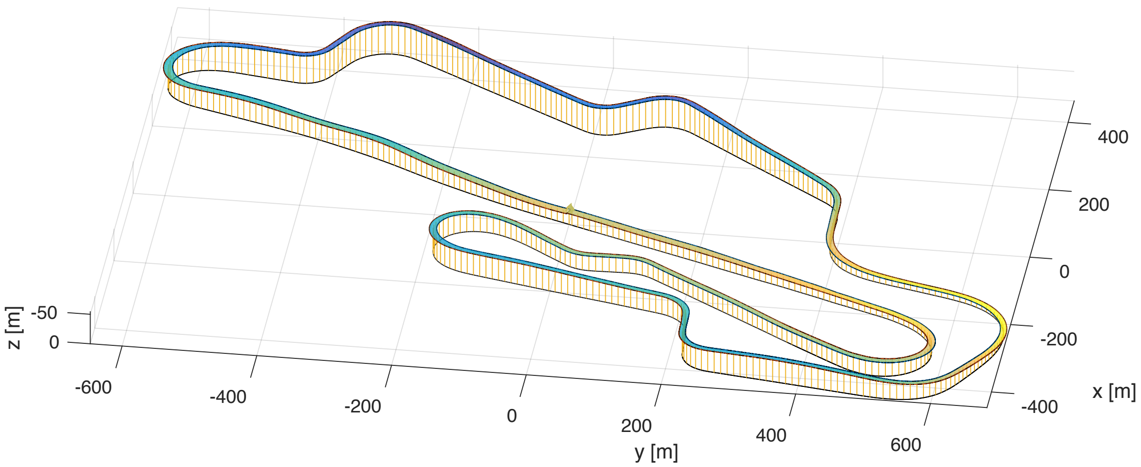


Figure 3.5: Overview of the $3D$ Mugello circuit.

Table 3.3: Parameters used in the track reconstruction problem.

variable	value
w_u	1
σ_{u_θ}	7.0×10^{-3}
σ_{u_σ}	2.4×10^{-5}
σ_{u_β}	2.4×10^{-5}
$\sigma_{u_{rw}}$	0.32
σ_{u_s}	0.20

has been used for all the four tracks[2].

Figure 3.5 gives an overview of the reconstructed Mugello circuit, where the elevation variations along the track are noticeable. The detail of the RACC chicane of Montmelò circuit is shown in figure 3.6. The reconstructed road borders (blue line for *Pins*, yellow for *GPOPS-II*) well matches the reference points (purple dots), moreover there is no noticeable difference in the solution provided by the two solvers. The resulting road curvature (bottom plot) highlights that the difference between the two solutions is approximately of the 1%.

The difference between the solutions obtained with the two solvers are relatively very small (up to ≈ 1%) and they are due to the different integration scheme adopted. Indeed, as previously said, in the track reconstruction problem there are no path constraints, which means that no penalty terms are used in the indirect approach OCP formulation. Therefore, the minimisation problem obtained through the indirect and direct approach is exactly the

[2] It means that one mesh point per meter has been used in *Pins*, and one mesh interval every two meters, with two collocation points per mesh interval have been used in *GPOPS-II*.

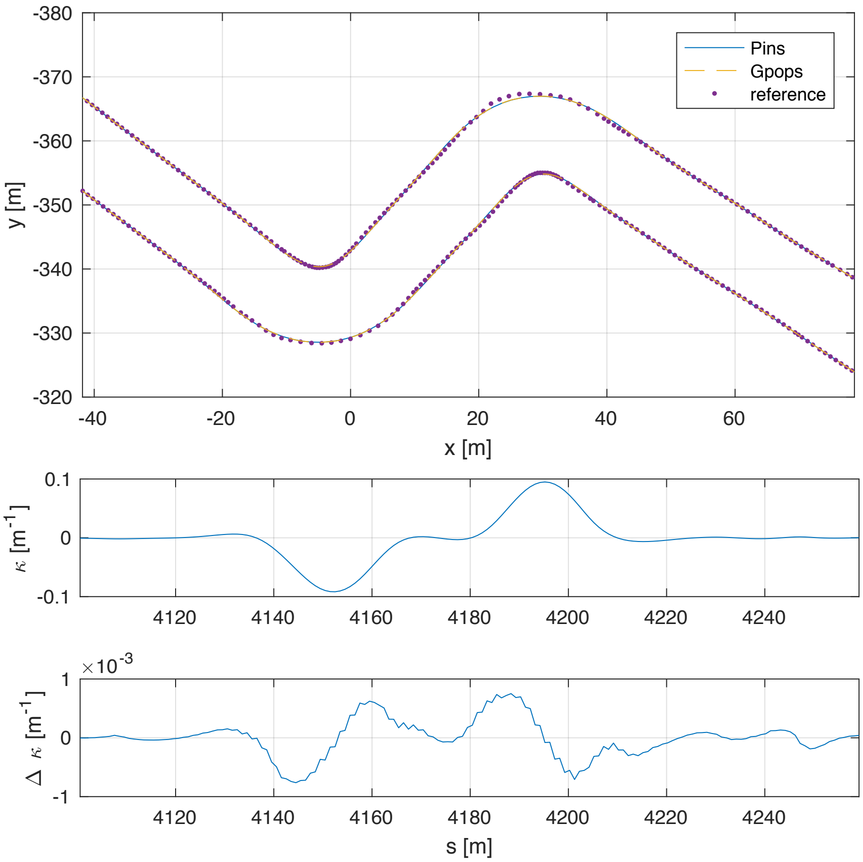


Figure 3.6: Detail of the RACC chicane of Montmelò circuit. In the top image, the road border reconstructed by *Pins* and *GPOPS-II* well matches the experimental data (purple dots) and not difference between the two solvers can be noticed by eye. The resulting road curvature (centre and bottom plots) shows that the difference between the two solutions is of approximately 1%.

Table 3.4: Summary of the maximum error from the reference track borders and the root mean square error for each circuit.

Circuit	*Pins*	*GPOPS-II*	*Pins* -*GPOPS-II* ratio
Maximum error			
Adria	2.02×10^0	2.01×10^0	1.00×10^0
Montmelò	1.50×10^0	1.54×10^0	9.69×10^{-1}
Mugello	9.01×10^{-1}	8.59×10^{-1}	1.05×10^0
Imola	8.99×10^{-1}	8.93×10^{-1}	1.01×10^0
Rms error			
Adria	4.01×10^{-3}	4.00×10^{-3}	1.00×10^0
Montmelò	2.51×10^{-3}	2.55×10^{-3}	9.87×10^{-1}
Mugello	1.79×10^{-3}	1.78×10^{-3}	1.00×10^0
Imola	1.93×10^{-3}	1.92×10^{-3}	1.00×10^0

same. The detail of the difference between the solutions provided by the two solvers is reported in table 3.4: for each circuit, the maximum error from the reference track borders and the root mean square of such error is calculated. Differences in the solutions are due to the different integration scheme.

Since the minimisation problem arising from the track reconstruction problem is exactly the same for *Pins* and *GPOPS-II*, it has been chosen as a test case for the robustness of the solver with respect to perturbation of the initial guess. The robustness has been tested through the following procedure:

1. A reference guess is first generated both for the states $\boldsymbol{x}^{\text{ref}}$ and controls $\boldsymbol{u}^{\text{ref}}$. The reference guess is characterised by the following state initialisation: $x_r^{\text{ref}} = x_{r0}$, $y_r^{\text{ref}} = y_{r0}$, $z_r^{\text{ref}} = z_{r0}$, $x_l^{\text{ref}} = x_{l0}$, $y_l^{\text{ref}} = y_{l0}$, $z_l^{\text{ref}} = z_{l0}$, $\theta^{\text{ref}} = \theta_0$ and $\kappa^{\text{ref}} = d\theta_0/d\zeta_0$. All other variables and controls are set to zero (i.e. $\boldsymbol{u}^{\text{ref}} = 0$);

2. for each state variable x, a relative noise amplitude, name a_x, is chosen. The relative noise amplitudes used are reported in table 3.5.

3. for each state variable x, its guess $x^{(g)}$ is given by the sum of the reference guess x^{ref} and a noisy term. The noisy term is the product of the relative noise amplitude a_x with a global noise variable ξ and a random variable r_x: $x^{(g)} = x^{\text{ref}} + a_x \xi r_x$. The global noise variable ξ is the same for all state variables, moreover the random variable r_x is uniformly distributed in the interval $[-1, 1]$ and sampled on each mesh point;

Table 3.5: Variable relative noise amplitudes used to generate the noisy guesses for the track reconstruction problem. Units are those of SI.

variable	x, y, z	r_w	θ, σ, β	$\hat{\theta}, \hat{\sigma}, \hat{\beta}$
relative amplitude	5	1	$\pi/10$	0

4. the global noise amplitude ξ is varied from 0 to 1.5 by step of 0.1, and ten different guesses are generated for each value of ξ, for a total of 150 different guesses;

5. the track reconstruction problem is then solved both with *Pins* and *GPOPS-II* using the noisy guesses. A maximum limit of 500 iterations is used; if the solver does not manage to solve the problem within this iteration limit, it is considered to fail.

The above described procedure has been repeated for each circuit. The results obtained are reported in figure 3.7, where the number of iterations required to solve the problem is shown as function of the noise global amplitude ξ. A number of iterations equal to the iteration limit (500) indicates the solver has not succeeded in finding the solution. The results highlight different trends for the $2D$ and $3D$ circuits. In $2D$ circuits (Adria and Montmelò) *GPOPS-II* is more robust and manages to solve the problem with all guesses, even at $\xi = 1.5$. A certain variance is observed in the number of iterations required to calculate the solutions for a given ξ, from approximately 20 iterations to 100. *Pins* fails to solve the problem for a noise amplitude grater than $\xi > 1$; however, for $\xi \leq 1$ it is able to calculate the solution in few iterations (≈ 6). The variance in the number of iterations for *Pins* is noticeably lower than that of *GPOPS-II*. In $3D$ circuits, *Pins* shows almost the same behaviour as with $2D$ circuits. *GPOPS-II* on the contrary presents a noticeably higher variability: while sometimes it still manages to solve the problem at $\xi = 1.5$, other times it fails even with a low relative noise ($\xi \approx 0.6$). On overall *Pins* demonstrates a more consistent behaviour, and *GPOPS-II* manages to handle guesses with an high noise but sometimes fails even with low-noise guesses. The greater robustness of *GPOPS-II* to less accurate guesses is probably due to the *IPOPT* initialisation procedure, and the heuristic therein used, to estimate the initial values for the Lagrange multipliers. On the contrary the current release of *Pins* does not implement any of such a procedure and sets the Lagrange initial values to zero. As a general result that authors have drawn there is that the Lagrange multiplier initialization is crucial for the robustness of both direct and indirect optimal control solvers.

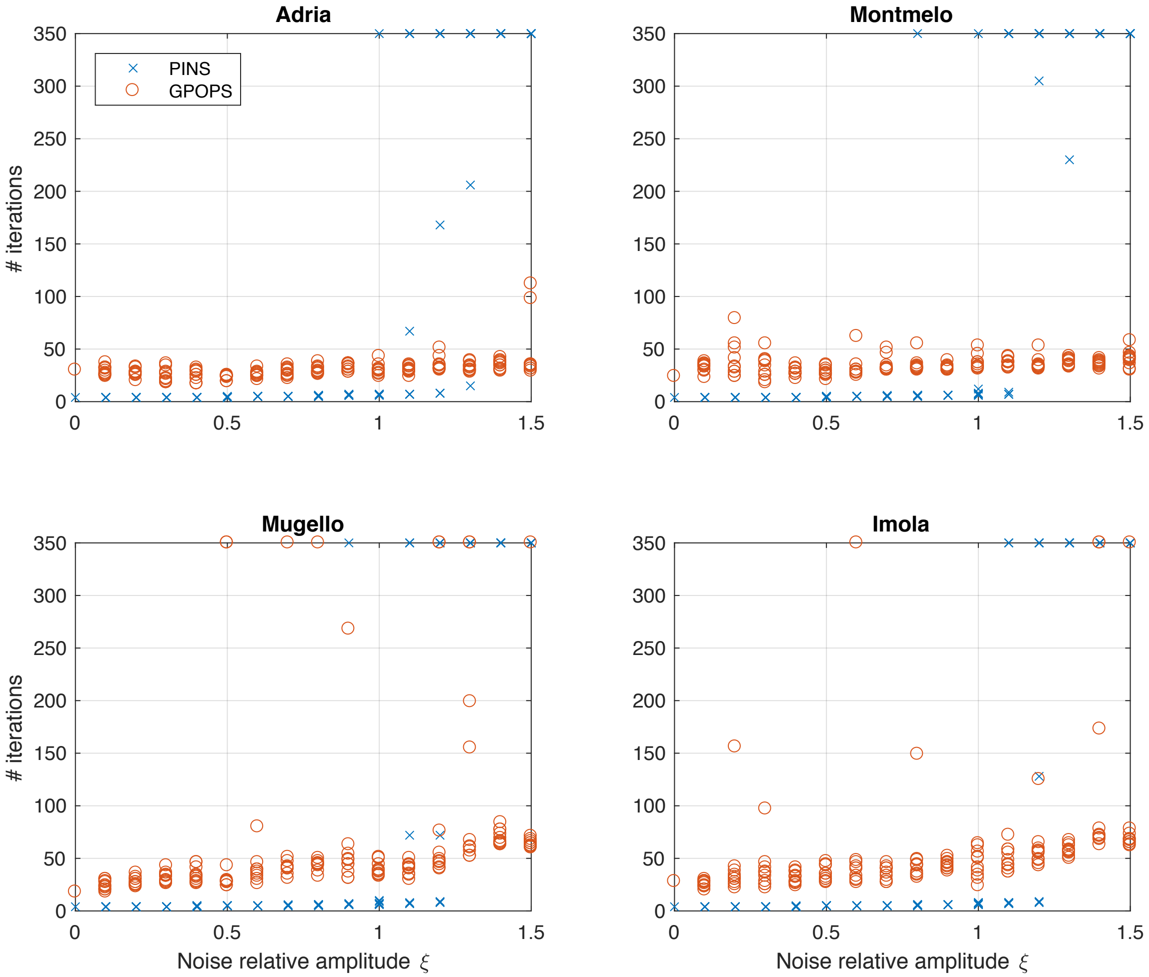


Figure 3.7: Robustness of the solvers with respect to noisy guess. The number of iterations required to compute the solution are plotted versus the relative noise amplitude. An iteration limit of 500 iterations has been chosen for both solvers. Points located at an ordinate of 500 iterations refer to non-converged problems.

3.3.3 Lap time simulation

The last OCP test that is analysed is the minimum lap time simulation of a racing GT car on the Adria International Raceway.

The simulation is performed with a 3-dof car model, which comprises quasi steady state load transfers and load-dependent tyre adherence. The three degrees of freedom of the car are: the speed V, the sideslip angle λ and the yaw rate Ω. The car position along the circuit is tracked by means of three variables: the curvilinear abscissa s, the lateral displacement from the road centre line n and the heading angle α w.r.t. the road centre line. The first order equations for the above mentioned six variables are:

$$\begin{aligned}
\dot{s} &= \frac{V\cos(\alpha-\lambda)}{1-n\kappa} \\
\dot{n} &= V\sin(\alpha-\lambda) \\
\dot{\alpha} &= \Omega - \frac{\kappa V\cos(\alpha-\lambda)}{1-n\kappa} \\
M\Omega V\lambda + M\dot{V} &= F_{tx}^{(rr)} + F_{tx}^{(rl)} + F_{tx}^{(fr)} + F_{tx}^{(fl)} - \delta(F^{(fr)} + F^{(fl)}) - F_d \\
M(\Omega V - \dot{V}\lambda - V\dot{\lambda}) &= \delta(F_{tx}^{(fr)} + F_{tx}^{(fl)}) + F_{ty}^{(rr)} + F_{ty}^{(rl)} + F_{ty}^{(fr)} + F_{ty}^{(fl)} \\
I_z z\dot{\Omega} &= a(F_{ty}^{(fr)} + F_{ty}^{(fl)}) - b(F_{ty}^{(rr)} + F_{ty}^{(rl)}) + t_v(-F_{tx}^{(rr)} + F_{tx}^{(rl)} - F_{tx}^{(fr)} + F_{tx}^{(fl)})
\end{aligned} \tag{3.10}$$

where M is the car mass, $F_{tx}^{(\hat{a}\hat{s})}$ is the tyre longitudinal force where $a = f, r$ indicates the front or rear axle, and $s = r, l$ indicates the right or left side, $F_{ty}^{(\hat{a}\hat{s})}$ is the tyre lateral force, δ is the steering angle, F_d is the drag force, $I_z z$ is the yaw inertia moment, a and b are respectively the distance of the front and rear axle form the centre of gravity and t_v is the car half width. In equations (3.10) the simplifications $\cos\xi \approx 1$ and $\sin\xi \approx \xi$ have been adopted for the sideslip λ and steering δ angles.

The drag force F_d is proportional to the square of the speed:

$$F_d = \frac{1}{2}\rho C_d V^2 \tag{3.11}$$

where ρ is the air density, C_d is the drag coefficient. Tyre lateral forces are given by a linear tyre model, while longitudinal ones are related to a single control u_x which is related to the normalized thrust:

$$\begin{aligned}
&F_{tx}^{(fl)} = F_{tx}^{(fr)} = \frac{Mg}{2} f^{-}(u_x)\gamma, \qquad F_{tx}^{(rl)} = F_{tx}^{(rr)} = \frac{Mg}{2}(f^{+}(u_x) + f^{-}(u_x)(1-\gamma)), \\
&F_{tx}^{(\hat{a}\hat{s})} = N^{(\hat{a}\hat{s})} K_{t\lambda} \lambda_t^{(\hat{a}\hat{s})}
\end{aligned} \tag{3.12}$$

where f^{+} and f^{-} return respectively the positive and negative part of the argument, γ is the front braking bias, $N^{(\hat{a}\hat{s})}$ is the tyre load, $\lambda_t^{(\hat{a}\hat{s})}$ is the tyre sideslip angle and $K_{t\lambda}$ the tyre sideslip stiffness. The tyre sideslip angles are given by the following expression:

$$\begin{aligned} \lambda_t^{(rr)} &= \lambda + \frac{\Omega(b+\lambda t_v)}{V} & \qquad \lambda_t^{(fr)} &= \lambda + \delta - \frac{\Omega(a-\lambda t_v)}{V} \\ \lambda_t^{(rl)} &= \lambda + \frac{\Omega(b-\lambda t_v)}{V} & \qquad \lambda_t^{(fr)} &= \lambda + \delta - \frac{\Omega(a+\lambda t_v)}{V} \end{aligned} \tag{3.13}$$

The tyre loads $N^{(\hat{a}\hat{s})}$ depend on the delayed vehicle longitudinal a_x and lateral a_y accelerations:

$$\begin{aligned} N^{(rr)} &= \frac{Mg}{4} + \frac{Mg}{4}\left(\frac{a-b+a_xh}{a+b} - a_y(1-\chi)\frac{h}{t_v}\right) \\ N^{(fr)} &= \frac{Mg}{4} + \frac{Mg}{4}\left(-\frac{a-b+a_xh}{a+b} - a_y\chi\frac{h}{t_v}\right) \\ N^{(rl)} &= \frac{Mg}{4} + \frac{Mg}{4}\left(\frac{a-b+a_xh}{a+b} + a_y(1-\chi)\frac{h}{t_v}\right) \\ N^{(fl)} &= \frac{Mg}{4} + \frac{Mg}{4}\left(-\frac{a-b+a_xh}{a+b} + a_y\chi\frac{h}{t_v}\right) \end{aligned} \tag{3.14}$$

where χ is the roll stiffness. The longitudinal a_x and lateral a_y accelerations follow the actual vehicle accelerations with a low band pass filter of time constant τ_{a_x} and τ_{a_y} in order to simulate the suspension load transfer lag:

$$\tau_{a_x}\dot{a}_x + a_x = \dot{V} + \Omega V\lambda \qquad \tau_{a_y}\dot{a}_y + a_y = \Omega V - (\dot{V\lambda}) \tag{3.15}$$

Summarising, the state space model comprises eight degrees of freedom $(V, \lambda, \Omega, s, n, \alpha, a_x, a_y)$, with the corresponding eight first order equations (3.10), (3.15) and two controls (u_x, δ).

OCP constraints

The minimum lap time problem includes some constraints, which ensures that the power used is less than the a maximum threshold, that the car never exceeds the track boundaries, and that the tyre forces are less than the tyre maximum adherence. Such constraints can be expressed as follows:

$$\begin{gathered} c_p = \frac{V(F_{tx}^{(rr)} + F_{tx}^{(rl)})}{P^{(max)}} \leq 1 \qquad c_n = \frac{n}{n_{\max}} \leq 1 \qquad c_n = \frac{-n}{-n_{\max}} \leq 1 \\ c_t^{(\hat{a}\hat{s})} = \left(\frac{F_{tx}^{(\hat{a}\hat{s})}}{N^{(\hat{a}\hat{s})}\mu_x^{(\hat{a}\hat{s})}}\right)^2 + \left(\frac{F_{ty}^{(\hat{a}\hat{s})}}{N^{(\hat{a}\hat{s})}\mu_y^{(\hat{a}\hat{s})}}\right)^2 \leq 1 \end{gathered} \tag{3.16}$$

where $P^{(max)}$ is the engine maximum power, $n_{\max}$ is the maximum lateral displacement which is equal to half of the road width r_w minus half of the car width $n_{\max} = r_w/2 - t_v$.

Table 3.6: Car dataset used for the minimum lap time problem.

variable	value	units	description
M	1184	kg	total mass
w	2.76	m	wheelbase
b	1.404	m	wheelbase to front axis distance
b	1.356	m	wheelbase to rear axis distance
t_v	0.807	m	half track width
h	0.4	m	CoM height from ground
$I_z z$	1775	kgm^2	yaw inertia moment
γ	0.62	-	braking bias
χ	0.5	-	roll stiffness
ρ	1.2	kg/m^3	air density
C_d	0.88	m^2	drag coefficient
μ_{x0}	1.68	-	tyre longitudinal adherence
μ_{y0}	1.68	-	tyre lateral adherence
K_μ	-0.5	-	tyre adherence variation with load
$K_{t\lambda}$	44	-	tyre lateral stiffness
τ_{a_y}	0.2	s	lateral load transfer time constant
τ_{a_x}	0.2	s	longitudinal load transfer time constant
$P^{(max)}$	215	kW	maximum power

$\mu_x^{(\hat{a}\hat{s})}$ and $\mu_y^{(\hat{a}\hat{s})}$ are respectively the tyre longitudinal and lateral adherence, which depend on the tyre loads:

$$\mu_x^{(\hat{a}\hat{s})} = \mu_{x0} + K_\mu \frac{N^{(\hat{a}\hat{s})}}{N_0^{(\hat{a}\hat{s})}} \qquad \mu_y^{(\hat{a}\hat{s})} = \mu_{y0} + K_\mu \frac{N^{(\hat{a}\hat{s})}}{N_0^{(\hat{a}\hat{s})}} \tag{3.17}$$

where $N_0^{(\hat{a}\hat{s})}$ is the tyre load in static conditions and K_μ is constant factor. The numerical data used to feed the car model is reported in table 3.6.

Solution analysis

The above described car model has been used to simulate the lap time of a GT car on the Adria International Raceway. A mesh with 1 discretisation point per meter has been used[3]. The lap time calculated by *Pins* and *GPOPS-II* are respectively $75.721s$ and $75.429s$, with a relative difference of the approximately 0.3%.

The simulated speed profile is shown in figure 3.8; the speed difference between the two solutions is always less than $3km/h$. In general, *Pins* simulated speed is higher than

[3] In *GPOPS-II* one mesh interval every two meters and two collocation points per mesh interval have been used.

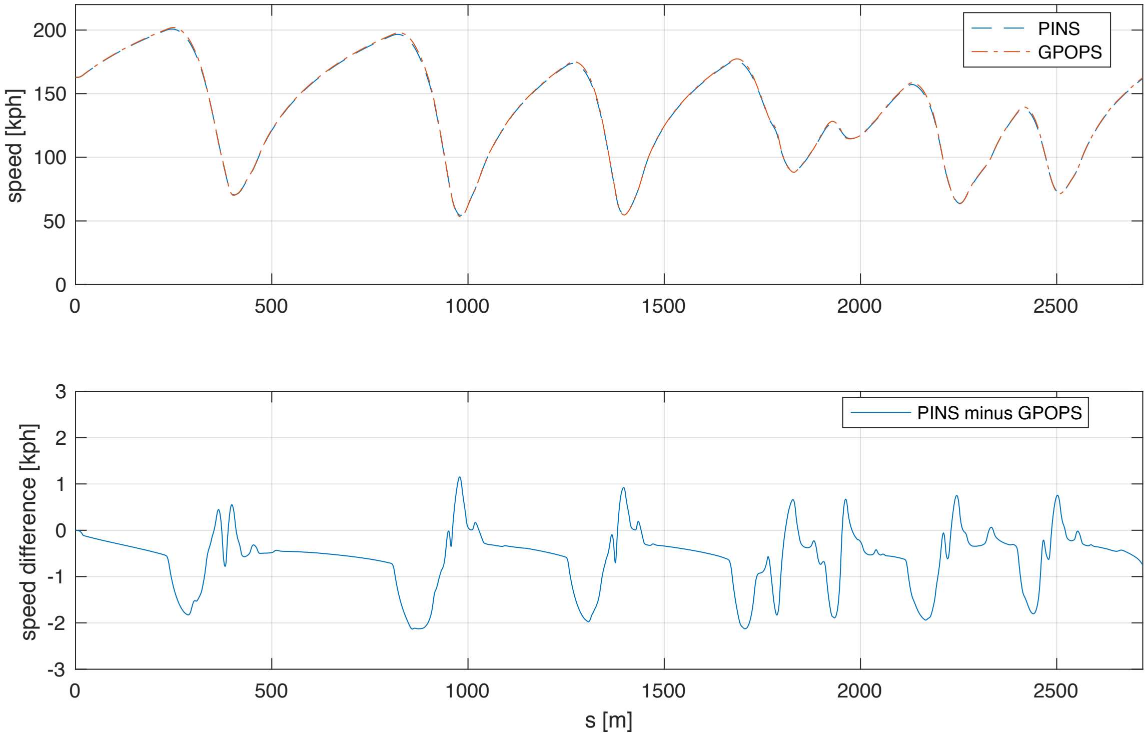


Figure 3.8: Simulated speed profile and speed difference (*Pins* minus *GPOPS-II*) along Adria circuit; the difference between the two solvers is always less than $3km/h$. In general, *Pins* simulated speed is higher than *GPOPS-II* one in the middle of the turns but it is lower in the straights and in the first part of the braking manoeuvre.

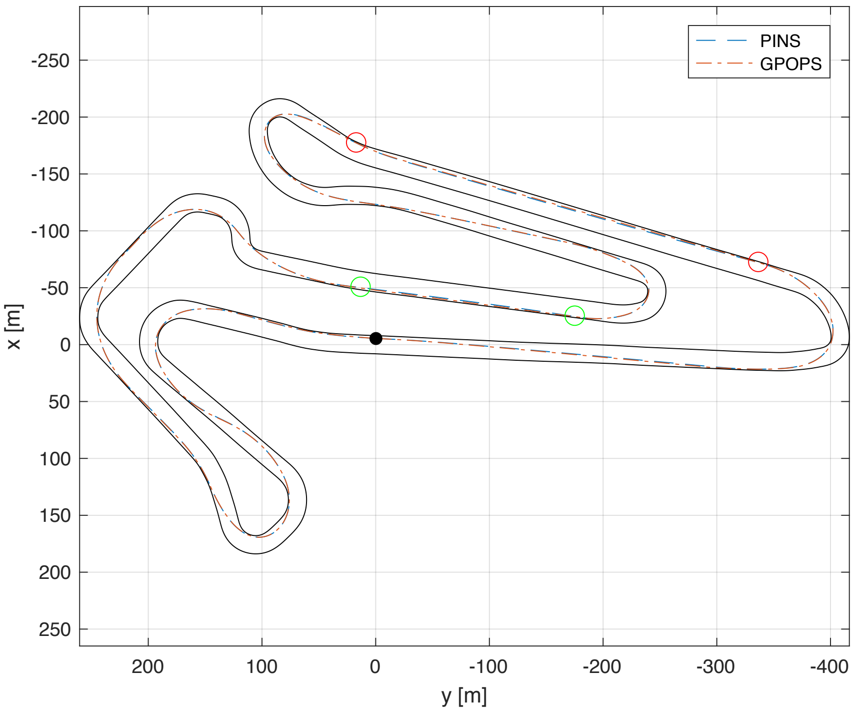


Figure 3.9: Simulated optimal trajectory along Adria circuit. The most significant trajectory difference is in the two intervals comprised between the red and green circles.

GPOPS-II one in the middle of the turns but it is lower in the straights and in the first part of the braking manoeuvres. The reason can be found in the use of the penalties, which in general do not allow to reach the control exact limits producing less braking/tractive forces. As a consequence the optimal solution found try to compensates maximising the speed in each corner looking for a slightly larger curvature radius.

The effect of the use of the penalties emerges when the optimal trajectories are compared (see figure 3.9). The two solutions show a good agreement along all the circuit, except in the two straights track sections comprised between the red circles (the first) and between the green ones (the second). The difference in the trajectory is more evident in the car lateral displacement n from the road centre line, which is reported in the top plot of figure 3.10. In the two mentioned track sections it is evident that *Pins* trajectory moves towards the road centre line, while *GPOPS-II* remains close to the track border in the two straights. In other words *Pins* tends to reduce the penalty (associated to the road border constraints, see (3.16)) by moving to the road centre line. It is opinion of the authors that this effect is probably evident in these two sections because there this manoeuvre affects only marginally the lap time.

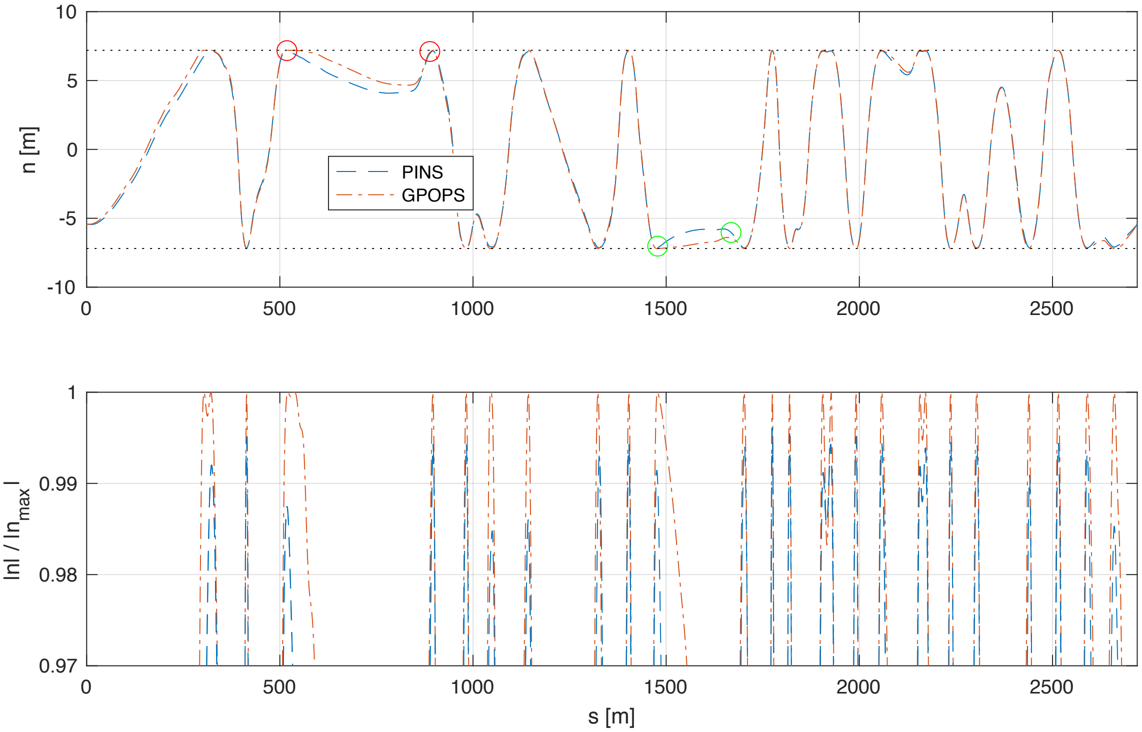


Figure 3.10: In the top figure, the lateral displacement n from the road centre line is shown. The trajectory differences in the two straight sections comprised between the red and green circles are noticeable. The bottom plot focuses on the road borders constraint enforcement $|c_n|$ (3.16) ($|c_n| = 1$ is the constraint limit). It can be noted that, while *GPOPS-II* solution arrives at a constraint value $|c_n| = 1$, *Pins* one does not go beyond $|c_n| = 0.995$.

The bottom plot of figure 3.10 shows the enforcement of the road borders constraint c_n (see (3.16)). It is possible to notice that, while *GPOPS-II* solution almost touches the road borders ($|c_n| = 1$), *Pins* one does not go beyond $|c_n| = 0.995$. The same behaviour can be observed also in the other OCP constraints: the maximum power limit and the tyre adherence limit. The former constraint is shown in figure 3.11; the bottom plot highlights that *Pins* solution uses up to $\approx 99\%$ of the maximum power, while *GPOPS-II* one uses up to 100% (at least to machine precision). Moreover, similar conclusions can be stated for the tyre engagement constraints $c_{t(\hat{a}\hat{s})}$ (3.16), which are reported in figure 3.12; in this case, *Pins* solution arrives up to $\approx 99.7\%$ of the maximum value.

The fact that *Pins* solution does not reach the exact constraints bounds (track width, engine power, tyre adherence) is a consequence of the penalty approach used by indirect methods. However even *IPOPT*, that is based on an Interior Point algorithm, uses a similar penalty approach [95]. The main difference between *Pins* and *IPOPT* is that the latter implements an algorithm that automatically sharpens the penalties in order to better better

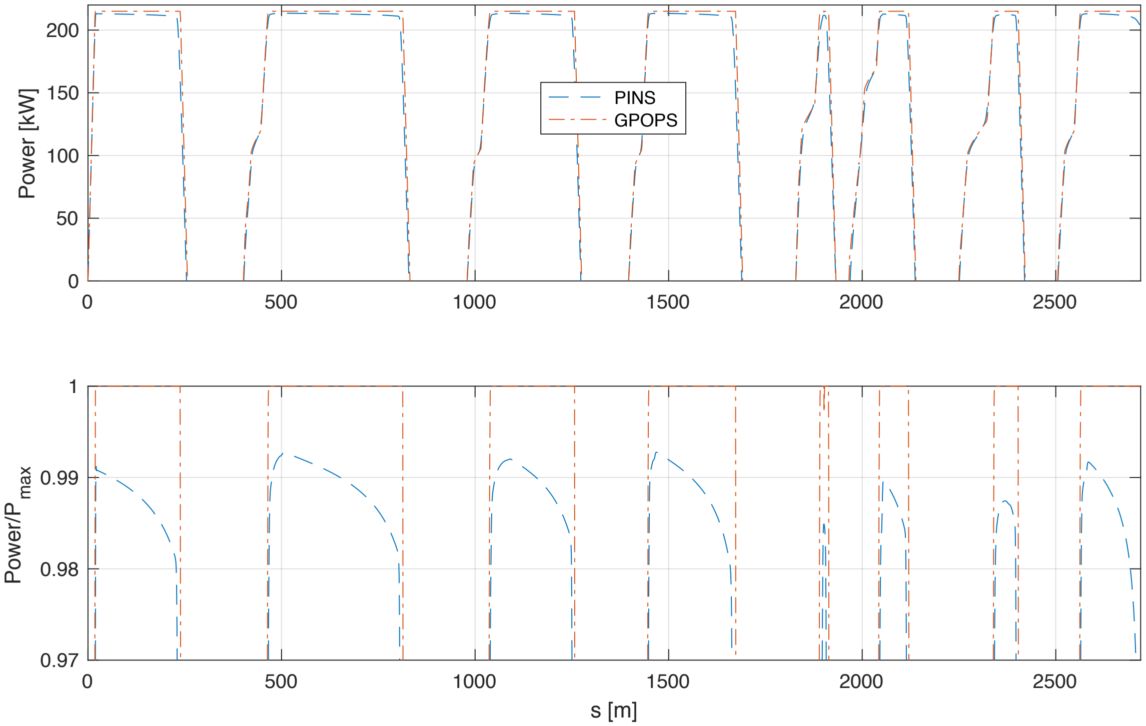


Figure 3.11: The power used $(S^{(rr)} + S^{(rl)})V$ is shown in the top plot; the two solutions present almost identical trends and both get close to the maximum power limit. The bottom plot focuses on the power-limit-constraint enforcement c_p (3.16) ($c_p = 1$ is the constraint limit). It is possible to notice that, while *GPOPS-II*arrives at $|c_p| = 1$, *Pins* arrives up to $c_p \approx 99\%$.

satisfy the constraints, while in the former the penalties can be fine-tuned only manually. Since in *Pins* this is achieved with a continuation procedure, it usually requires longer computational time and diminishes the robustness to convergence using a non optimal tuning strategy. The values that have been used are, according to authors' opinion, is the best compromise between constraint enforcement, robustness of the solver and low computational times.

At the beginning of this section it has been stated that *Pins* simulated lap time is approximately $0.3s$ higher than *GPOPS-II* one, then the results showed that this difference can be attributed to the lower engine power and tyre engagement usage of *Pins* solution. While $0.3s$ may seem a consistent discrepancy from an engineering point of view, it should be remembered that the absolute performance resulting from lap time simulations is not of primary importance because it depends on some parameters that are difficult to measure, first of all the friction between tyres and asphalt. Usually, the numeric dataset used to feed the mathematical model is tuned so as to make the simulated car performance match

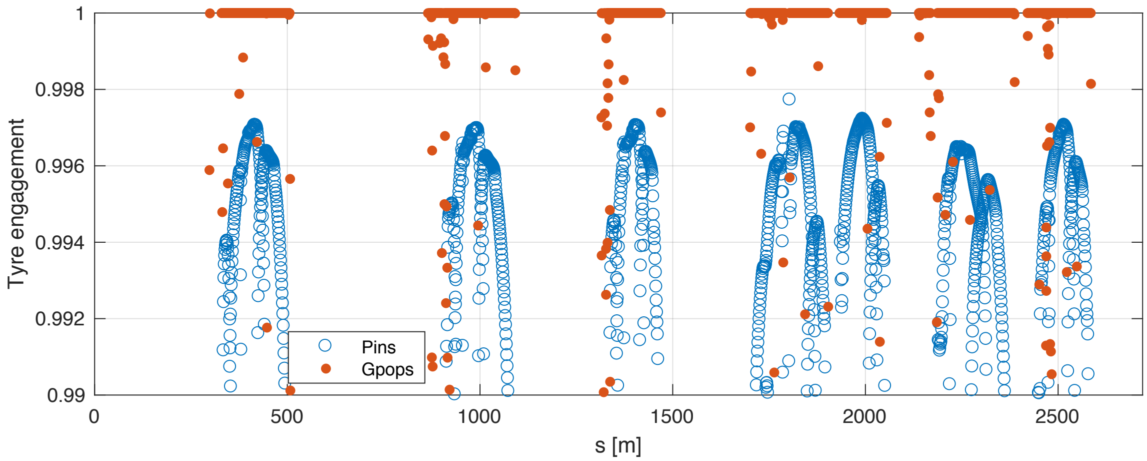


Figure 3.12: The tyre engagement $c_{t(\hat{a}\hat{s})}$ (3.16) of the four wheels are shown as function of the curvilinear abscissa. Again, *GPOPS-II*solution touches the constraint limit, while *Pins* arrives up to the $\approx 99.7\%$ of the maximum value.

the telemetry data. Only after this calibration process is performed, the model is used for simulations and optimisation. Therefore, in lap time simulations the sensitivity to model parameter variations is much more important than the absolute lap time.

The sensitivity of the two solver has been compared on the optimisation of the braking bias γ and roll stiffness χ. Simulations have been performed varying the braking bias and the roll stiffness respectively in the range $[0.55, 0.68]$ and $[0.63, 0.8]$; the lap times differences between *Pins* and *GPOPS-II* solutions are shown in figure 3.13. One may note that, while the difference in the simulated lap time varies in the range $[0.29s, 0.32s]$, the location of the best lap time does not differs relevantly between the two solvers. In particular *Pins* minimum lap time is achieved with a braking bias of $\gamma = 0.595$ and a roll stiffness of $\chi = 0.741$ (with an accuracy of 0.001 both for γ and χ), while *GPOPS-II* minimum lap time is obtained for the same value of the rolling stiffness but for a braking bias slightly higher, $\gamma = 0.597$. This discrepancy in the location of optimum value of the parameters is mainly due to two different causes. The first is the different simulated performance: since *Pins* solution uses slightly less engine power and tyre adherence, this results in a different car dynamics and thus optimal braking bias and roll stiffness. The second cause is the control regularisation induced by the control penalties[4] that makes the controls' solution to be smoother, thus modifying again the simulated car dynamics. In order to verify that the different location of the best lap time is due to these two causes, the *Pins* model has been modified by increasing the engine maximum power and the tyre adherence and by

[4] Control penalties are used in the indirect approach to keep the controls bounded.

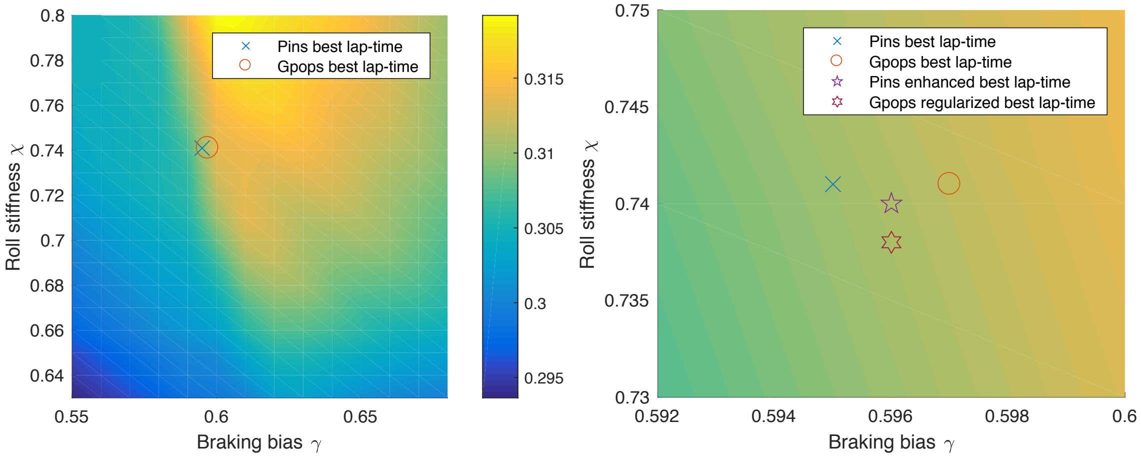


Figure 3.13: The difference of the simulated lap time (*Pins* minus *GPOPS-II*) is shown as function of the braking bias γ (x-axis) and roll stiffness χ (y-axis). The location of the best lap time is also shown. The right plot focuses on a smaller range of the variables γ and χ in order to better highlight the differences in the location of the best lap time. The best lap time time for the *Pins* "enhanced" and *GPOPS-II* "regularized" models are also shown.

augmenting the track width, so as to make the constraints actually reach their physical limits. In particular, the engine maximum power has been increased by a 1.015 factor, the tyre adherence by 1.004, and the track width by 1.01, in accordance with the distance from the constraints boundaries highlighted in figures 3.10, 3.11 and 3.12. This model, referred to as *Pins* "enhanced" model, shows a best lap time for $\gamma = 0.596$ and $\chi = 0.74$, thus it is located closer to *GPOPS-II* minimum lap time but differences are still present. Finally *GPOPS-II* model has been regularised by adding a small regularisation term proportional (1×10^2) to the square of the controls into the Lagrange target; this model is referred to as *GPOPS-II* "regularised". The regularisation is not exactly the same as that of *Pins*, yet tries to mimic it. The optimum parameters for the *GPOPS-II* regularised model are $\gamma = 0.596$ and $\chi = 0.738$, as shown in the right plot in figure 3.13. These results shows that the car design optimisations performed by *Pins* and *GPOPS-II* are in agreement if the same car performance is simulated and similar regularisation is adopted.

The robustness of the two methods with respect to perturbation of the initial guess has been studied with the same procedure described in section 3.3.2. Differently from section 3.3.2, the minimum lap time OCP includes constraints that are treated with different approaches by direct and indirect methods. The reference (i.e. without noise) guess is characterised by a non-zero guess speed $V_0 = 30kph$, while all the other variables are left to zero. The noise relative amplitudes a_x for each state variable are reported in table 3.7, and the noise global amplitude ξ spans from 0 to 1 by steps of 0.1. Moreover, the speed

Table 3.7: Variable relative amplitudes used to generate the noisy guesses for the minimum lap time problem. Units are those of SI.

variable	n	α	V	λ	Ω	ax	ay	u_x	δ
relative amplitude	7	0.6	60	0.05	1	10	15	1	0.15

V is forced to be greater than $1m/s$: when the noised speed results to be less than such threshold, it is set to $1m/s$. This allows to avoid numerical singularities in the first order equations, which can not be made explicit ODE in the curvilinear abscissa domain s when $V = 0$. The solver iterations limit is now increased to 10^3 since the problem is more difficult to solve.

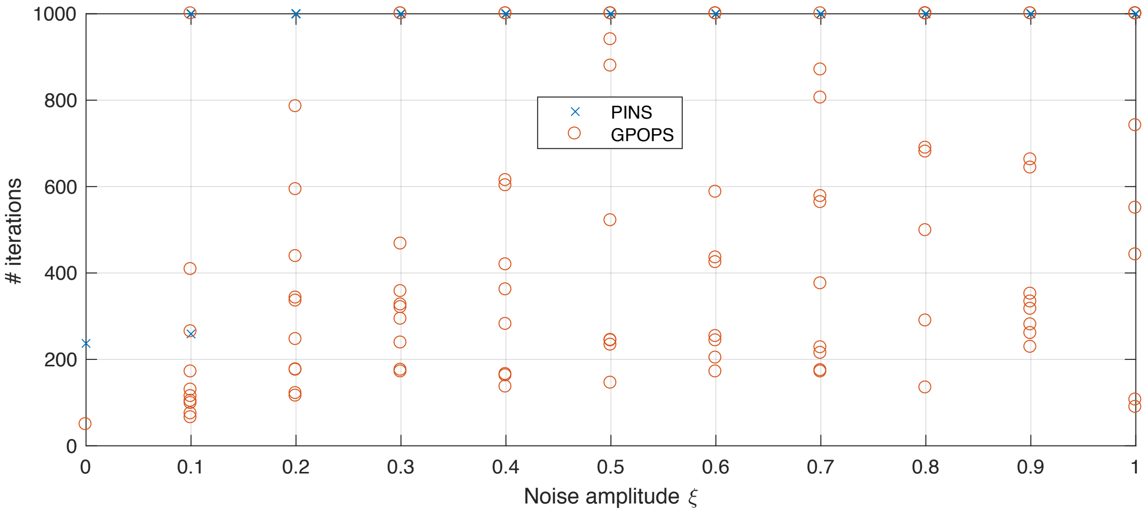


Figure 3.14: Robustness of the solvers with respect to noisy guess. The number of iterations required to compute the solution are plotted versus the relative noise amplitude. An iteration limit of 10^3 iterations has been chosen for both solvers. Points located at an ordinate of 10^3 iterations refer to non-converged problems.

The results obtained are summarised in figure 3.14, and are significantly different from those of the reconstruction problem in section 3.3.2. Even with a low noise amplitude $\xi = 0.1$ *Pins* fails to find the solution most of the times, and from $\xi = 0.3$ it is never successful. *GPOPS-II*, like in section 3.3.2, shows a more variable behaviour and for all values of ξ it sometimes manages to find the solution, sometimes it fails. None of the two solvers appear to be robust with respect to the added noise. It is opinion of the authors that the high failure rate highlighted by *Pins* is consequence of the constraints violation in the provided guess. When starting from a noisy guess, *Pins* showed very high residual since the first iterations and then it straggle to find a feasible (i.e. with low residual) region. High values for the residual ($\approx 10^6$) are due to high penalty values (i.e. constraints violation), which make the problem highly ill conditioned and difficult to solve. Differently *IPOPT*, the NLP solver used for *GPOPS-II*, implements a feature that project the current state into the constraint-feasible region, thus even if the guess violates the constraints it automatically project it into a feasible region [95]. It is opinion of the authors that this is the main advantage that makes *GPOPS-II* more robust than *Pins* with respect to less accurate initial guesses.

3.4 Result discussion and notes on performance

The results presented in the previous section, and in particular those of the basic motorcycle problem, confirmed that *GPOPS-II* implements a more accurate integration scheme (Gaussian in *GPOPS-II*, midpoint in *Pins*) that is able to provide a significantly higher accuracy than *Pins* using relatively few mesh points. Despite this, *Pins* is able to reach the absolute highest accuracy when using an higher number of mesh points.

Moreover *GPOPS-II* demonstrated to better handle perturbations (i.e. noise) in the guess, in particular when constraints are present in the OCP. Indeed, while *Pins* showed similar robustness results in the track reconstruction problem, it largely under performed in the minimum lap time one. It is opinion of the authors that this is mainly due to the robustness of *IPOPT* that implements a mechanism that project the variables into the feasible region, and adopts a reliable initialisation of the Lagrange multipliers. On the contrary, *Pins* uses a zero guess for the Lagrange multipliers, moreover, if the guess violates the constraints, it suffers of ill-conditioning due to the high penalty values (since constraints are not projected into a feasible region like in *IPOPT*).

In the minimum time problem, the constraint enforcement analysis showed that *Pins* solution does not reach the exact constraints boundaries (as *GPOPS-II* does). In particular, its solution did not make full usage of the track width, engine power and tyre adherence. This behaviour is a direct consequence of the lack of automatic tuning of the penalty weights and parameters during the convergence to the solution.

Finally, when tested on a typical utilisation scenario (i.e. optimisation of car design parameters) the two software provided similar outcomes. In particular, the location of the optimal braking bias and roll stiffness given by *Pins* and *GPOPS-II* were very close, with relative differences of the order of 10^{-3}. The small differences could be explained by the slightly different simulated car dynamics due to different constraint usage and regularisation.

In all previous test problems nothing has been stated about the solver performance related to the time required to compute the solution. A comparison of such performance for the two algorithm is not an easy task because the two solvers are developed with differ-

Table 3.8: Solution computing times measured on a desktop computer equipped with an Intel Xeon E3-1270 v5 processor with 32GB of ram, running on Ubuntu 16.04. *GPOPS-II* has always been used with *IPOPT* as NLP solver and the ma57 linear solver[34]. *Pins* turns out to be significantly faster than *GPOPS-II* in most of the problems.

Problem	Number of mesh points	*Pins* time [s]	*GPOPS-II* time [s]	*GPOPS-II* to *Pins* time ratio
Motorcycle basic	100	0.2	0.9	4.5
Motorcycle basic	5000	7	2	0.3
Adria 2D	2719 (1 per meter)	0.1	9	90
Montmelo 2D	4650 (1 per meter)	0.1	11	110
Mugello 3D	5244 (1 per meter)	0.5	60	120
Imola 3D	4906 (1 per meter)	0.5	35	70
Car model	2719 (1 per meter)	6	50	8
Car model	54380 (20 per meter)	96	932	10
Car model	543800 (200 per meter)	1184	10180	8.5

ent programming languages. *Pins* is entirely written in *c++* and, even if it uses a Ruby interpreter to setup the problem data, the solution algorithm and the problem function evaluation are computed by compiled code. *GPOPS-II* instead is developed as a *MATLAB*© library, but uses a compiled NLP solver (a compiled version of *IPOPT* has been used here). Thus, function evaluation is done at *MATLAB*© level (even if the user may compile the functions into mex files) while the NLP solution is calculated by compiled code. Thus a comparison of the solvers performance is not really meaningful to compare the efficiency of the algorithms, but it is certainly interesting from a practical (i.e. user) point of view. In the test problem analysed in this work, it has been noticed that *Pins* is generally significantly faster than *GPOPS-II*; the measured computing times are reported in table 3.8.

3.5 Summary

In this chapter indirect and direct optimal control methods have been compared on three test problems related to vehicle optimal control applications. Numerical solvers *Pins* and *GPOPS-II* have been chosen respectively as representative for the indirect and direct approach since they are the software most used in literature for this purpose. The results showed that each software has advantages and disadvantages compared to the other. In particular, *Pins* excelled in computational time and absolute accuracy, while *GPOPS-II* resulted to be more robust and, when using coarse meshes, even more accurate. Moreover, from a user point of view, *GPOPS-II* may seem more user-friendly since it does not require any fine-tuning of the inequality constraints, which are automatically managed by the NLP solver. *Pins* instead requires the user to properly adjust the penalty parameters in order to achieve the best performance. While this is slightly time consuming, it offers a deeper control on the solution strategy and allows the user to choose whether to prefer fast solution computing or accurate constraint enforcement. However, from a general perspective, it can be stated that initialisation phase of the optimal control problems plays a major role in solver robustness. On overall results showed that indirect and direct methods on overall have a similar behaviour when dealing with minimum lap time problems; most of the observed differences between the two solvers can be explained with different numerical implementation features rather than with intrinsic differences of indirect and direct methods. Such implementation features include: integration scheme, multiplier initialisation, penalty tuning algorithm, projection of the state into feasible region.

Chapter 4

Modelling and lap time simulations of four-wheeled vehicles

In this chapter optimal control theory is applied to minimum lap time problems of four-wheeled vehicles. Particular attention is dedicated to the modelling of the vehicles, since different modelling choices affect both simulation outcomes and the solving capabilities of numerical software. The relevance of different modelling choices for the simulation outcomes is first investigated, then a novel multibody model of a GP2 car for optimal control problems is proposed and used for lap time simulations on the Montmelò circuit. Finally, similar modelling and simulation techniques are applied to go-karts.

4.1 Introduction

Since early attempts to solve minimum lap time problems in the late 50's [82], huge improvements have been done in minimum lap time problems, and in the last decades they have become widely used to simulate the performance of race cars and motorbikes, and to optimise their design.

Vehicle modelling is a crucial element in lap time simulations, especially in optimal control ones. The vehicle model should be able to capture the essence of the dynamics so as to lead to reliable simulations, yet at the same time it should not be too complex — e.g. with several dof — otherwise the resulting optimal control problem may be practically non solvable. A good vehicle model thus must include all relevant dynamics features (for the specific case of study) and at the same time it must lead to a practically solvable, with current numerical solvers, optimal control problem. Since optimal control numerical solvers are faster and more robust when the Jacobian and Hessian matrices are calculated exactly [20, 11] (that means not through numerical approximations), the exact derivatives of the vehicle equations of motion should be known. Thus, well-known car models of commercial multibody software, like *ADAMS*, *Virtual Lab Motion*, *CarSim*, *RecurDyn* and *SimPack* are of little use in optimal control problems. Mathematical modelling is instead the natural way to develop a vehicle model where the equations of motion, and their derivatives, are known. All the vehicle models presented in this chapter have been developed with the mathematical approach, with use of the symbolic algebra software *Maple*© and the *MBSymba* library [60]. The models are carefully described so as to makes it clear which dynamic features are included and which are disregarded. A possible different approach for the vehicle modelling may be the use of software (as those based on *Modelica*) that allows the user to only declare bodies and interactions between them; then the software automatically generates a code for the evaluation of the equations of motion and their derivatives. It is opinion of the author that the mathematical approach is favourable over the latter approach since it gives the possibility to read, really understand, and manipulate the equations of motions. A deep understanding of the equations makes it clear which physical phenomena are included, moreover it allows to modify the equations by simplifying those terms that are uselessly complex and may cause numerical issues. On the other hand this approach requires more expert developer.

Looking at the historical development of optimal control lap time simulations, it can be noted that the complexity of vehicle models has increased in parallel with the improvements of numerical solvers and computer processing power. Car models in the first 00's [21, 22, 23] used to include seven dof: three dof for the chassis gross motion (longitudinal, lateral and yaw motion), and four dof for the wheel spins. Tyre loads were calculated in quasi-steady-state (QSS) conditions by assuming a perfect balance equations of the vertical forces and of the pitch and roll torques acting on the chassis. Thus the chassis motion inertia was neglected, moreover a constant ratio between the lateral tyre load transfer at the front axle and at the rear one was assumed. Since then car models with seven dof has been widely used [48, 72, 65, 58, 92]. In some applications substantial new features and improvements are introduced: in [21] and [49] a model that included suspension travel is presented; even though suspensions are modelled as simple vertical springs (i.e. suspension kinematics is not taken into account), tyre loads are dynamically calculated and chassis inertia is considered. Moreover, the same author in [49] adds a thermodynamic tyre model. In [58] the basic seven dof car model is endowed with an energy recovery system (simulating a Formula 1), in [56] a three dimensional [1] track model is introduced and in [65] the aerodynamic-suspension interaction of a Formula 1 car is modelled through a quasi steady state pre-calculated map.

[1] The term "three-dimensional" road is here used improperly since the road actually is a two-dimensional manifold. Three-dimensional means that the road is not flat and it extends on a three-dimensional space, i.e. the road presents elevation variations.

Sometimes even simpler single track models are used, as in [90, 91]; however these two works do not aim at solving lap time problems, but at demonstrating the advantage of the handbrake technique in particular scenarios.

This brief but comprehensive survey of car models for optimal control lap time simulations shows that despite the model complexity has continuously increased with time, these simulations still almost always rely on a relatively simple seven dof model (three dof for the chassis gross motion and four for the wheel spins). While the car models with three degrees of freedom for the chassis gross motion (and QSS tyre loads) lead to a simplification of the resulting optimal control problem, it is not clear what level of accuracy in the simulation outcomes these simplified models may provide. In particular these models do not take into account for the suspension-related loads transfer delay, moreover a constant roll stiffness ratio between the front and rear axle is assumed, which may be far from reality. Lap time simulations are often used to optimise the vehicle design and setup, therefore simulation outcomes should not be negatively affected by simplifications in the vehicle model. It is a priori not clear what are the most important vehicle dynamics features that should be included for accurate results. Multiple choices for the desired model structure can be made and different approximations can be assumed; common simplifications usually affect wheels, suspensions and/or chassis dynamics. Even controls, that simulate driver inputs, can be inserted into the model at different levels of the dynamic. For instance, traction control can relies, in order of complexity, in tyre forces, wheels slip, or axle torque. Generally, the highest modelling complexity is adopted for the particular dynamic that has to be investigated, while reasonable simplifications are taken in the rest of the model. However, a deep understanding of consequences of such approximations is still missing in literature, and how they affect simulation results is yet not well quantifiable.

The first part of this chapter tries to give answer to this question, with a comparison of three car models with different levels of detail in the car dynamics. The aim is to understand how different models can influence both simulation outcomes and computational cost of lap time simulation: it is not a priory clear which complexity should be adopted to get the best compromise between reliable results and short computational times. In the second part of this chapter (section 4.3) a state-of-the art model for a GP2 car is presented. This model aims to improve existing car models for optimal control simulations by including effects related to suspension kinematics and a more accurate aerodynamic interaction where drag and lift forces depend on the dynamically calculated ride heights. Lap time simulation outcomes of this model will be validated through comparison with experimental data recorded during a qualifying lap in the 2012 season. Finally, the third part of this chapter (section 4.4) is dedicated to lap time simulation of a racing go-kart.

The peculiar characteristic of go-karts is the lack of rear axle differential, which leads to a totally different design strategy from race cars. Section 4.4 is motivated not only by the particular dynamics resulting from the absence of differential, but also by the lack in literature of examples of go-kart lap time simulations.

4.2 The significance of high-order dynamics in lap time simulations

The aim of this section is to understand how different car models can influence both minimum lap time simulations outcomes and the computational cost of the optimal control problem. Three car models with different levels of detail in the car dynamics modelling are here developed and then compared in minimum lap time simulations and car design optimisation.

In the first part of this section the three car models, that are a 14 dof GT class model together with other two simpler (ten and seven dof) models, with will be described. The full features model (14 dof) is composed by the car chassis which is a rigid bodies with six dof (translation and rotations in the three space dimensions), plus four wheels with two additional dof each, respectively the suspension motion and wheel spin. The second model has been derived by assuming the suspensions to be in quasi steady state conditions, i.e. on each wheel the suspension force is always balanced by the tyre load. This allows to remove the four suspension-related dof yet at the same time the chassis motion is not neglected and the tyre loads are not calculated in quasi steady state conditions. Finally, the third model is the classically used one with seven dof, i.e. the chassis roll, pitch and vertical motion with respect to the road plane are neglected and tyre loads are the QSS ones.

In the last part of this section the simulations outcomes obtained with these models will be compared. Since lap time simulations are often used for the design and setup optimisation of vehicles, the sensitivity of the three models to variations of the centre of mass position and of the suspension stiffness will be analysed. Simulations are performed on the Adria International Raceway.

4.2.1 Full features dynamic car model (14 dof)

The full features multibody model is composed by the chassis body that can translate and rotate about the three Cartesian axes, plus the four wheels that can spin and move vertically due to the suspension motion. The model thus comprises 14 degrees of freedom, as reported in table 4.1: six are related to the chassis motion, plus two additional dof for each wheel, that are the vertical displacement and the wheel spin. Tyres loads are dynamically calculated

Table 4.1: Degrees of freedom of the 14 dof car model.

degrees of freedom	units	description
s	m	position along road curvilinear abscissa
n	m	lateral displacement from road middle lane
z	m	CoM vertical displacement from the nominal height
ϕ	rad	chassis roll angle
μ	rad	chassis pitch angle
α	rad	chassis yaw angle relative to the road middle lane direction
$z_w^{(\hat{a}\hat{s})}$	m	wheel vertical travel
$\omega_w^{(\hat{a}\hat{s})}$	rad/s	wheel spin

through tyres radial stiffness; tyre longitudinal and lateral forces are calculated as function of the loads and slip quantities through a Pacejka magic formula. Aerodynamic drag and down-force have been taken into account too. Controls rely in steering angle and wheel torques: positive torques are exerted only on the rear axle while negative ones (braking) are divided between front and rear wheels with a constant bias. A differential with torque distribution has been implemented in the rear axle.

The road model used here is the $2D$ model described in appendix A; the $2D$ road model has been chosen instead of the $3D$ one because the Adria circuit does not present any noticeable elevation variations. With reference to the road coordinate system described in appendix A, the relative frame $\boldsymbol{W}_{rel}$ that describes the motion of the vehicle chassis with respect to (w.r.t) the road middle lane position is given by the following transformation sequence: a lateral translation of n, a rotation about the z-axis of α, a vertical displacement of $z - h$ (h is the nominal CoM height from ground), a rotation about the x-axis of ϕ and finally rotation about the y-axis of μ. Thus, the vehicle (chassis) frame $\boldsymbol{W}_v$ is equal to:

$$\boldsymbol{W}_v = \boldsymbol{W}_r\boldsymbol{W}_{rel} = \boldsymbol{W}_r\boldsymbol{T}(0,n,0)\boldsymbol{R}_z(\alpha)\boldsymbol{T}(0,0,z-h)\boldsymbol{R}_x(\phi)\boldsymbol{R}_y(\mu) \tag{4.1}$$

where $\boldsymbol{W}_r$ is the road reference system. It turns out the n is the lateral displacement from the road middle lane, α is the relative heading w.r.t the road middle lane, z is the CoM vertical displacement w.r.t the nominal position, ϕ and μ are respectively the roll and pitch angles[2]. It is recalled that SAE convention is adopted for the axes orientation. Since wheel vertical travels of GT cars is less than $\approx 40mm$, wheel vertical travels may be considered small[3], and consequently chassis vertical displacement z, roll ϕ and pitch angles μ are small,

[2] It should be noted that the meaning of z, ϕ and μ is slightly different from the same quantities defined in section 6.4 for the motorbike model. Here the translation about the z-axis of the quantity $z-h$ is performed before the roll rotation of the angle ϕ about the x-axis.

[3] Wheel vertical travels can be considered *small* if compared to the overall car dimensions, track width

i.e. every function of μ, ϕ and z has been approximated with its Taylor series expansion up to the first order. The resulting chassis relative reference system $\boldsymbol{W}_{rel}$ therefore is:

$$\begin{aligned}\boldsymbol{W}_{rel} &= \big(\boldsymbol{T}(0,n,0)\boldsymbol{R}_z(\alpha)\big)\,\big(\boldsymbol{T}(0,0,z-h)\boldsymbol{R}_x(\phi)\boldsymbol{R}_y(\mu)\big)\\ &= \begin{bmatrix}\cos\alpha & -\sin\alpha & 0 & 0\\ \sin\alpha & \cos\alpha & 0 & n\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 & \mu & 0\\ 0 & 1 & -\phi & 0\\ -\mu & \phi & 1 & z-h\\ 0 & 0 & 0 & 1\end{bmatrix}\end{aligned} \tag{4.2}$$

The equations of motion for the full vehicle are calculated within a frame $\boldsymbol{W}_{v0}$ that follows only the lateral displacement and the relative heading of the vehicle:

$$\begin{aligned}\boldsymbol{W}_{v0} &= \boldsymbol{W}_r\boldsymbol{T}(0,n,0)\boldsymbol{R}_z(\alpha)\\ &= \boldsymbol{W}_r\begin{bmatrix}\cos\alpha & -\sin\alpha & 0 & 0\\ \sin\alpha & \cos\alpha & 0 & n\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}\end{aligned} \tag{4.3}$$

With this definition of $\boldsymbol{W}_{v0}$, the vehicle chassis frame $\boldsymbol{W}_v$ can be written as $\boldsymbol{W}_v = \boldsymbol{W}_{v0}\boldsymbol{T}(0,0,z-h)\boldsymbol{R}_x(\phi)\boldsymbol{R}_y(\mu)$. Moreover the frame $\boldsymbol{W}_{v0}$, that is shown in figure 4.1, has the same speed V, sideslip angle λ and yaw rate Ω of the car chassis. These three quantities (V, λ, Ω) are used as state variables.

Newton equations of motion are calculated within the $\boldsymbol{W}_{v0}$ frame, and, taking into account all the external forces acting on the vehicle (aerodynamic and tyre forces), they are:

$$m\left(\dot{V} + V\Omega\lambda\right) = \sum_{\hat{a}\hat{s}} F_{tx}^{(\hat{a}\hat{s})} - (F_{ty}^{(fr)} + F_{ty}^{(fl)})\delta - F_d \tag{4.4a}$$

$$m\left(V\Omega - V\dot{\lambda} - \dot{V}\lambda\right) = \sum_{\hat{a}\hat{s}} F_{ty}^{(\hat{a}\hat{s})} + F_{tx}^{(fl)}\delta + F_{tx}^{(fr)}\delta \tag{4.4b}$$

$$m(\ddot{z} - g) + \sum_{\hat{a}\hat{s}} m_w^{(\hat{a})}\ddot{z}_w^{(\hat{a}\hat{s})} = -\sum_{\hat{a}\hat{s}} N^{(\hat{a}\hat{s})} - F_l \tag{4.4c}$$

where $F_{tx}^{(\hat{a}\hat{s})}, F_{ty}^{(\hat{a}\hat{s})}, N^{(\hat{a}\hat{s})}$ are respectively the longitudinal, lateral and normal tyres forces on each wheel, F_d, F_l are the aerodynamic drag and down-force, moreover $z_w^{(\hat{a}\hat{s})}$[4] are the

and wheelbase, which are respectively of $\approx 1.5m$ and $\approx 3m$. As rough calculation, the ratio between the maximum positive travel at front minus the maximum negative travel at rear and the car track width is of ≈ 0.05, which indeed is small.

[4] with $\hat{a} \in \{r, f\}$ indicating the rear or front axle and $\hat{s} \in \{r, l\}$ indicating the right or left side.

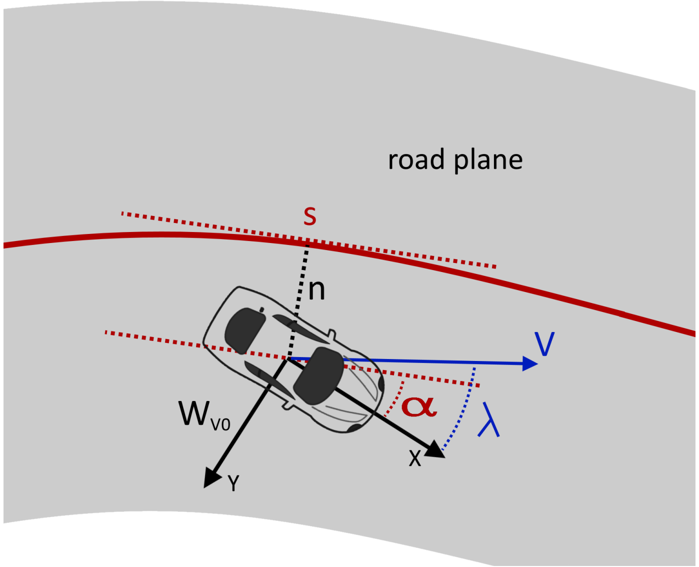


Figure 4.1: Representation of the frame $\boldsymbol{W}_{v0}$ with respect to the road. The x-y plane of such frame lays on the road plane. The lateral displacement n and the relative yaw angle α from the road middle lane are highlighted. The vehicle speed V and the sideslip angle λ are shown too.

wheel vertical[5] travels with respect to the nominal position (positive when the suspension spring gets compressed), $\omega_w^{(\hat{a}\hat{s})}$ are the wheel spin velocities (positive as the vehicle speed) and $m_w^{(\hat{a})}$ are wheel masses. For sake of simplicity the complete suspension kinematics is not taken into consideration and the wheels can move only along the z axis of the chassis frame $\boldsymbol{W}_v$. In all the equations of motion the time derivatives of z, ϕ and μ, together with the drift λ and steering δ angles, have been considered small too.

Euler's equations calculated w.r.t the origin of the frame $\boldsymbol{W}_{v0}$ are the following:

$$\begin{aligned}
I_{xx}\ddot{\phi} + \left(I_{zz}\mu - I_{xx}\mu - I_{xz} - \sum_{\hat{a}\hat{s}} m_w^{(\hat{a})} b^{(\hat{a})} z_w^{(\hat{a}\hat{s})} \right) \dot{\Omega} + (I_{zz} - I_{xx} - I_{yy})\Omega\dot{\mu} \\
-M\left(hV\dot{\lambda} + h\lambda\dot{V} + (z-h)\Omega V \right) - \sum_{\hat{a}\hat{s}} m_w^{(\hat{a})} t_v^{(\hat{a}\hat{s})} \ddot{z}_w^{(\hat{a}\hat{s})} \\
- \left((I_{yy} - I_{zz})\phi + \sum_{\hat{a}\hat{s}} m_w^{(\hat{a})} t_v^{(\hat{a}\hat{s})} z_w^{(\hat{a}\hat{s})} \right) \Omega^2 - \sum_{\hat{a}\hat{s}} m_w^{(\hat{a})} z_w^{(\hat{a}\hat{s})} \Omega V \\
+ I_w^{(f)}(\dot{\omega}_w^{(fr)} + \dot{\omega}_w^{(fl)})\delta + I_w^{(f)}(\omega_w^{(fr)} + \omega_w^{(fl)})\dot{\delta} + \Omega \sum_{\hat{a}\hat{s}} I_w^{(\hat{a})} \omega_w^{(\hat{a}\hat{s})} = \tau_x
\end{aligned} \tag{4.5a}$$

[5] "Vertical" means along the z axis of the chassis frame $\boldsymbol{W}_v$.

$$\begin{aligned}
&I_{yy}\ddot{\mu} + \left((I_{yy}-I_{zz})\phi + \sum_{\hat{a}\hat{s}} m_w^{(\hat{a})} t_v^{(\hat{a}\hat{s})} z_w^{(\hat{a}\hat{s})}\right)\dot{\Omega} + (I_{xx}+I_{yy}-I_{zz})\Omega\dot{\phi} \\
&\quad +M\left((z-h)\dot{V} - h\lambda\Omega V\right) + \sum_{\hat{a}\hat{s}} m_w^{(\hat{a})} b^{(\hat{a})} z_w^{(\hat{a}\hat{s})} + \sum_{\hat{a}\hat{s}} m_w^{(\hat{a})} z_w^{(\hat{a}\hat{s})}\dot{V} \\
&\quad -\sum_{\hat{a}\hat{s}} I_w^{(\hat{a}\hat{s})}\dot{\omega}_w^{(\hat{a}\hat{s})} + \left((I_{zz}-I_{xx})\mu - I_{xz} + \sum_{\hat{a}\hat{s}} m_w^{(\hat{a})} b^{(\hat{a})} z_w^{(\hat{a}\hat{s})}\right)\Omega^2 \\
&\qquad +I_w^{(f)}\Omega(\dot{\omega}_w^{(fr)} + \dot{\omega}_w^{(fl)})\delta + I_w^{(f)}\Omega(\omega_w^{(fr)} + \omega_w^{(fl)})\dot{\delta} = \tau_y
\end{aligned} \tag{4.5b}$$

$$(I_{zz}+2I_{xz}\mu)\dot{\Omega} - \sum_{\hat{a}\hat{s}} I_w^{(\hat{a})}\omega_w^{(\hat{a}\hat{s})}\dot{\phi} - I_{xz}(\ddot{\phi} - 2\Omega\dot{\mu}) - \phi\sum_{\hat{a}\hat{s}} I_w^{(\hat{a}\hat{s})}\dot{\omega}_w^{(\hat{a}\hat{s})} = \tau_z \tag{4.5c}$$

where I_{xx}, I_{yy}, I_{zz} are the principal moments of inertia of the car chassis, I_{xz} is the cross moment of inertia, $I_w^{(r)}$ and $I_w^{(f)}$ are the rear and front wheel spin inertia moment, $t_v^{(r)} = t_v^{(rl)} = -t_v^{(rr)}$ is the rear half track, $t_v^{(f)} = t_v^{(fl)} = -t_v^{(fl)}$ is the front half track, $b^{(r)} = b$ is the x-axis distance of the car CoM from the rear axle, and $b^{(f)} = -a$ is the x-axis distance of the car CoM from the front axle. Moreover τ_x, τ_y, and τ_z are respectively the x, y and z component of the net external torque acting on the chassis:

$$\tau_x = \sum_{\hat{a}\hat{s}} N^{(\hat{a}\hat{s})}(t_v^{(\hat{a}\hat{s})} + h\phi) \tag{4.6a}$$

$$\tau_y = F_l x_A - F_d y_A + \sum_{\hat{a}\hat{s}} N^{(\hat{a}\hat{s})}\left(-b^{(\hat{a})} + (h - r_t^{(\hat{a})})\mu\right) \tag{4.6b}$$

$$\begin{aligned}
\tau_z = &\sum_{\hat{a}\hat{s}} F_{tx}^{(\hat{a}\hat{s})}(t_v^{(\hat{a}\hat{s})} + h\phi) + a\left(F_{tx}^{(fr)} + F_{tx}^{(fl)}\right)\delta \\
&+\sum_{\hat{a}\hat{s}} F_{ty}^{(\hat{a}\hat{s})}\left((h - r_t^{(\hat{a})})\mu - b^{(\hat{a})}\right) + \left(F_{tx}^{(fr)}t_v^{(f)} - F_{tx}^{(fl)}t_v^{(f)}\right)\delta
\end{aligned} \tag{4.6c}$$

where r_t is the tyre radius and x_A, y_A are respectively the x-distance from the CoM and the height from the road plane of the aerodynamic centre of pressure. It is worth pointing out that the choice of calculating the Euler equations with respect to the origin of $\boldsymbol{W}_{v0}$ simplifies the roll (4.5a) and pitch (4.5b) equations by avoiding the presence of any term related to longitudinal or lateral tyre forces.

The equations governing the vertical dynamics of the wheels are the following:

$$m_w^{(\hat{a})}\left(\ddot{z}_w^{(\hat{a}\hat{s})} - g + \ddot{z} + \ddot{\mu}b^{(\hat{a})} - \ddot{\phi}t_v^{(\hat{a}\hat{s})}\right) = -N^{(\hat{a}\hat{s})} - F_s^{(\hat{a}\hat{s})} \tag{4.7}$$

where $F_s^{(\hat{a}\hat{s})}$ represent the suspension equivalent vertical force applied from the wheels to the chassis, i.e. a compressed suspension corresponds to a negative force. $F_s^{(\hat{a}\hat{s})}$ is the sum

Table 4.2: Vehicle parameters

symbol	value	units	description
b	1.12	m	distance between the rear axle and the vehicle CoM
a	1.48	m	distance between the front axle and the vehicle CoM
$w = a + b$	2.6	m	wheelbase
h	0.4	m	CoM height
$2t_v^{(f)}$	1.71	m	front track
$2t_v^{(r)}$	1.62	m	rear track
m	1400	kg	total mass
I_{xx}	400	kgm^2	x-axis vehicle inertia (w.r.t. CoM)
I_{yy}	2000	kgm^2	y-axis vehicle inertia (w.r.t. CoM)
I_{zz}	1320	kgm^2	z-axis vehicle inertia (w.r.t. CoM)
I_{xz}	0	kgm^2	mixed x-z-axis vehicle inertia (w.r.t. CoM)
$m_w^{(f)}$	15	kg	front wheel unsprung mass
$m_w^{(r)}$	17	kg	rear wheel unsprung mass
$I_w^{(f)}$	1.1	kgm^2	front wheel spin inertia
$I_w^{(r)}$	1.3	kgm^2	rear wheel spin inertia
$r_t^{(f)} = r_t^{(r)}$	0.29	m	nominal tyre radius
$r_{tr}^{(f)} = r_{tr}^{(r)}$	0.29	m	tyre rolling radius
$K_{ss}^{(f)}$	200×10^3	N/m	front spring stiffness
$K_{ss}^{(r)}$	180×10^3	N/m	rear spring stiffness
$K_a^{(f)}$	140×10^3	N/m	front anti-roll bar stiffness
$K_a^{(r)}$	120×10^3	N/m	rear anti-roll bar stiffness
$K_{es}^{(f)} = K_{es}^{(r)}$	200×10^3	N/m	tyres radial stiffness
$K_{td}^{(f)} = K_{td}^{(r)}$	150	Ns/m	tyres radial damping
c_{dA}	0.34	-	aerodynamics drag coefficient
c_{lA}	0	-	aerodynamics lift coefficient
γ	0.7	-	braking bias
k_d	100	Nms/rad	differential stiffness
N_0	5000	N	tyre nominal vertical load
$\mu_{x0}^{(f)}$	1.57	-	front tyre max longitudinal adherence at nominal load
$\mu_{x0}^{(f)}$	1.56	-	rear tyre max longitudinal adherence at nominal load
$\mu_{y0}^{(f)}$	1.41	-	front tyre max lateral adherence at nominal load
$\mu_{y0}^{(f)}$	1.51	-	rear tyre max lateral adherence at nominal load
$K_{t\lambda 0}^{(f)}$	25	rad^{-1}	front tyre sideslip stiffness at nominal load
$K_{t\lambda 0}^{(f)}$	31	rad^{-1}	rear tyre sideslip stiffness at nominal load
$K_{t\kappa 0}^{(f)}$	34	-	front tyre longitudinal stiffness at nominal load
$K_{t\kappa 0}^{(f)}$	32	-	rear tyre longitudinal stiffness at nominal load

of the spring, damper and anti-roll bar and is given by:

$$\begin{aligned} F_s^{(\hat{a}r)} &= F_{s0}^{(\hat{a}r)} + K_{ss}^{(\hat{a})} z_w^{(\hat{a}r)} + K_{sd}^{(\hat{a})} \dot{z}_w^{(\hat{a}r)} + K_a^{(\hat{a})} (z_w^{(\hat{a}r)} - z_w^{(\hat{a}l)}) \\ F_s^{(\hat{a}l)} &= F_{s0}^{(\hat{a}l)} + K_{ss}^{(\hat{a})} z_w^{(\hat{a}l)} + K_{sd}^{(\hat{a})} \dot{z}_w^{(\hat{a}l)} - K_a^{(\hat{a})} (z_w^{(\hat{a}r)} - z_w^{(\hat{a}l)}) \end{aligned} \tag{4.8}$$

where $K_{ss}^{(\hat{a})}$, $K_{ss}^{(\hat{a})}$, $K_a^{(\hat{a})}$ are respectively the (rear or front) spring, damper and anti-roll bar stiffness (reduced at the wheel), and $F_{s0}^{(\hat{a}\hat{s})}$ are the suspension forces at the nominal car trim (i.e. the load on the wheel minus the wheel weight).

The wheel spin dynamic equations are:

$$I_w^{(\hat{a}\hat{s})} \dot{\omega}_w^{(\hat{a}\hat{s})} = \tau_{wd}^{(\hat{a}\hat{s})} - F_{tx}^{(\hat{a}\hat{s})} \left(r_t^{(\hat{a}\hat{s})} - \xi^{(\hat{a}\hat{s})} \right) \tag{4.9}$$

where $I_{w\theta}$ is the wheel spin inertia, $\tau_{\theta\eta}$ is the overall torque acting on the wheel, $r_t^{(\hat{a}\hat{s})}$ is the tyre nominal radius, and $\xi^{(\hat{a}\hat{s})}$ is the tyre radial deformation. It can be noted that the expression multiplying the longitudinal tyre force is the loaded radius of the tyre. The tyre deformations $\xi^{(\hat{a}\hat{s})}$ can be calculated through simple kinematic expressions:

$$\xi^{(\hat{a}\hat{s})} = \mu b^{(\hat{a})} - \phi t_v^{(\hat{a}\hat{s})} + z_w^{(\hat{a}\hat{s})} + z \tag{4.10}$$

The four torque inputs $\tau_{wd}^{(\hat{a}\hat{s})}$ are not independent each other but depend on one variable, the overall driving torque τ_d, which is the sum of the engine (positive part, τ_e) and braking (negative part, τ_b) torque. Moreover the simultaneous presence of a driving torque at the rear axle and a braking torque at the front one is not allowed:

$$\tau_d = f^+(\tau_d) + f^-(\tau_d) \equiv \tau_e + \tau_b \tag{4.11}$$

where f^- and f^+ return respectively the (regularized) negative and positive part of the argument. The traction torque τ_e is delivered only to the rear axle, while the braking one τ_b is split between both axes with a constant front braking bias γ. Moreover, at the rear axle a velocity sensitive differential is present. Thus, the torque delivered to each wheel is:

$$\begin{aligned} \tau_{wd}^{(rr)} &= \frac{\tau_e}{2} + \tau_e k_d (\omega_w^{(rl)} - \omega_w^{(rr)}) + (1-\gamma)\frac{\tau_b}{2} \\ \tau_{wd}^{(rl)} &= \frac{\tau_e}{2} - \tau_e k_d (\omega_w^{(rl)} - \omega_w^{(rr)}) + (1-\gamma)\frac{\tau_b}{2} \\ \tau_{wd}^{(fr)} &= \gamma\frac{\tau_b}{2} \\ \tau_{wd}^{(fl)} &= \gamma\frac{\tau_b}{2} \end{aligned} \tag{4.12}$$

where k_d is the differential stiffness.

Tyre model

Tyre vertical loads may be easily calculated as a function of tyre radial deformations, and for simplicity it is assumed that the elastic and damping forces are linear:

$$N^{(\hat{a}\hat{s})} = f^{+}\left(N_0^{(\hat{a}\hat{s})} + K_{tr}^{(\hat{a})}\xi^{(\hat{a}\hat{s})} + K_{td}^{(\hat{a})}\dot{\xi}^{(\hat{a}\hat{s})}\right) \tag{4.13}$$

where $N_0^{(\hat{a}\hat{s})}$ is the static tyre load, K_{tr} is the tyre radial stiffness, K_{td} is the tyre damping coefficient and ξ is the deformation given by equation (4.10). The f^{+} function prevents the loads to become negative when the wheel lifts.

Longitudinal and lateral tyre forces have been calculated as function of the tyre load N, the sideslip λ_t and longitudinal slip κ_t according to the Magic Formula Tyre model [70]. In particular, sideslip angles $\lambda_t^{(\hat{a}\hat{s})}$ can be calculated as follows:

$$\begin{aligned}
\lambda_t^{(rr)} &= \arctan\left(\frac{V\lambda + \Omega b}{V - \Omega t_v^{(r)}}\right) & \lambda_t^{(rl)} &= \arctan\left(\frac{V\lambda + \Omega b}{V + \Omega t_v^{(r)}}\right) \\
\lambda_t^{(fr)} &= \delta + \arctan\left(\frac{V\lambda - \Omega a}{V - \Omega t_v^{(f)}}\right) & \lambda_t^{(fl)} &= \delta + \arctan\left(\frac{V\lambda - \Omega a}{V + \Omega t_v^{(f)}}\right)
\end{aligned} \tag{4.14}$$

Tyre longitudinal slips are instead given by these relationships:

$$\begin{aligned}
\kappa_t^{(rr)} &= \frac{\omega_w^{(rr)} r_{tr}^{(r)}}{V - \Omega t_v^{(r)}} - 1 & \kappa_t^{(rl)} &= \frac{\omega_w^{(rl)} r_{tr}^{(r)}}{V + \Omega t_v^{(r)}} - 1 \\
\kappa_t^{(rl)} &= \frac{\omega_w^{(fr)} r_{tr}^{(f)}}{V + \Omega t_v^{(f)}} - 1 & \kappa_t^{(rl)} &= \frac{\omega_w^{(fl)} r_{tr}^{(f)}}{V + \Omega t_v^{(f)}} - 1
\end{aligned} \tag{4.15}$$

where r_{tr} is the tyre rolling radius. Finally, the time lag which is present in the generation of lateral forces due to lateral compliance of tyre carcass is modelled as a first order relaxation equation [70] for each tyre:

$$\begin{aligned}
\frac{\sigma_{tx}^{(r)}}{V - \Omega t_v^{(r)}}\dot{F}_{tx}^{(rr)} + F_{tx}^{(rr)} &= F_{tx,magic}^{(r)}\left(N^{(rr)}, \lambda_t^{(rr)}, \kappa_t^{(rr)}\right) \\
\frac{\sigma_{tx}^{(r)}}{V + \Omega t_v^{(r)}}\dot{F}_{tx}^{(rl)} + F_{tx}^{(rl)} &= F_{tx,magic}^{(r)}\left(N^{(rl)}, \lambda_t^{(rl)}, \kappa_t^{(rl)}\right) \\
\frac{\sigma_{tx}^{(f)}}{V - \Omega t_v^{(f)}}\dot{F}_{tx}^{(fr)} + F_{tx}^{(fr)} &= F_{tx,magic}^{(f)}\left(N^{(fr)}, \lambda_t^{(fr)}, \kappa_t^{(fr)}\right) \\
\frac{\sigma_{tx}^{(f)}}{V + \Omega t_v^{(f)}}\dot{F}_{tx}^{(fl)} + F_{tx}^{(fl)} &= F_{tx,magic}^{(f)}\left(N^{(fr)}, \lambda_t^{(fl)}, \kappa_t^{(fl)}\right)
\end{aligned} \tag{4.16}$$

where $\sigma_{tx}^{(r)}$, $\sigma_{tx}^{(f)}$ are rear and front tyre longitudinal relaxation lengths. Similar expressions hold for the lateral forces:

$$\begin{aligned}
\frac{\sigma_{ty}^{(r)}}{V-\Omega t_v^{(r)}}\dot{F}_{ty}^{(rr)}+F_{ty}^{(rr)} &= F_{ty,magic}^{(r)}\left(N^{(rr)},\lambda_t^{(rr)},\kappa_t^{(rr)}\right)\\
\frac{\sigma_{ty}^{(r)}}{V+\Omega t_v^{(r)}}\dot{F}_{ty}^{(rl)}+F_{ty}^{(rl)} &= F_{ty,magic}^{(r)}\left(N^{(rl)},\lambda_t^{(rl)},\kappa_t^{(rl)}\right)\\
\frac{\sigma_{ty}^{(f)}}{V-\Omega t_v^{(f)}}\dot{F}_{ty}^{(fr)}+F_{ty}^{(fr)} &= F_{ty,magic}^{(f)}\left(N^{(fr)},\lambda_t^{(fr)},\kappa_t^{(fr)}\right)\\
\frac{\sigma_{ty}^{(f)}}{V+\Omega t_v^{(f)}}\dot{F}_{ty}^{(fl)}+F_{ty}^{(fl)} &= F_{ty,magic}^{(f)}\left(N^{(fr)},\lambda_t^{(fl)},\kappa_t^{(fl)}\right)
\end{aligned} \tag{4.17}$$

where $F_{tx,magic}^{(r)}$, $F_{tx,magic}^{(r)}$, $F_{ty,magic}^{(r)}$ and $F_{ty,magic}^{(f)}$ are the functions returning respectively the rear longitudinal, front longitudinal, rear lateral and front lateral tyre forces (front and rear tyres have different parameters), accordingly to the the Pacejka Magic Formula Tyre Model [70]. The dependence of the tyre forces on the tyre roll angle has been neglected. It is opinion of the author that this simplification does not affect remarkably the results since, as it will be shown in the next sections, this model is intended to study tyre load dynamics, not tyre longitudinal or lateral forces.

State space

Chassis, wheels and tyre equations may be reduced to a set of first order equations by introducing auxiliary variables for the relevant chassis and suspension speeds:

$$\dot{z}=z_{dot} \qquad \dot{\phi}=\phi_{dot} \qquad \dot{\mu}=\mu_{dot} \qquad \dot{z}_w^{(\hat{a}\hat{s})}=z_{w,dot}^{(\hat{a}\hat{s})} \tag{4.18}$$

In conclusion, the state space variable vector $\boldsymbol{x}_{14}$ for the 14 dof model includes the following 29 variables:

$$\boldsymbol{x}_{14}=\left[V;\ \lambda;\ \Omega;\ z;\ \phi;\ \mu;\ z_{dot};\ \phi_{dot};\ \mu_{dot};\ \omega_w^{(\hat{a}\hat{s})};\ z_w^{(\hat{a}\hat{s})};\ z_{w,dot}^{(\hat{a}\hat{s})};\ F_{ty}^{(\hat{a}\hat{s})};\ F_{tx}^{(\hat{a}\hat{s})}\right] \tag{4.19}$$

The control input vector $\boldsymbol{u}$ has 2 variables:

$$\boldsymbol{u}=\left[\tau_d;\ \delta\right] \tag{4.20}$$

Table 4.3: Car natural frequencies

	full (14 dof)	massless-wheels (10 dof)	basic (7 dof)
weave	2.7 Hz	2.7 Hz	2.6 Hz
bounce	2.6 Hz	2.6 Hz	–
roll	5.1 Hz	4.9 Hz	–
pitch	3.2 Hz	3.1 Hz	–
wheels hop	$20 \div 30$ Hz	–	–
wheels spin	45 Hz	45 Hz	45 Hz

Chassis, wheels and tyre equations, after the reduction to the first order, can be put in the form:

$$\boldsymbol{A}_{14}(\boldsymbol{x}_{14})\dot{\boldsymbol{x}}_{14} = \boldsymbol{f}_{14}\left(\boldsymbol{x}_{14}, \boldsymbol{u}, t\right) \tag{4.21}$$

All numeric data used to feed the model is listed in table 4.2.

Natural frequencies

The natural frequencies of the model so far described are reported in table 4.3, where they are calculated at a fixed speed of $V = 30\ m/s$ and in straight motion $\Omega = 0$. Natural frequencies of the other two models that will be described in the next subsections are shown too. It can be noticed that the wheel spin is the fastest dynamics, while the wheel hop is slower, chassis motions and weave are the lowest. The higher the frequencies are, the more mesh points need to be used to simulate the dynamics, which produce a larger problem and numerical ill conditioning could occur. In different words, higher frequencies lead to problems that are more difficult to solve. A first option to reduce this high-frequency-related drawback would be to neglect the wheel spin dynamics, i.e. to assume $I_w^{(r)} = I_w^{(f)} = 0$. However, in order to calculate longitudinal tyre forces it is still necessary to keep spin rates $\omega_w^{(\hat{a}\hat{s})}$ as state variables. In other words, wheel spin equations (4.9) cannot be eliminated and have to remain as nonlinear algebraic equations. However, such equations cannot be exactly solved and substituted into the remaining equations; moreover the optimal control software used here to solve the minimum lap time problem (*Pins*, see chapter 3), only recently introduced the support for algebraic constraints of index one. At the time this study was performed, it strictly required an ODE problem formulation so wheel spin equations has to be included in the state space. Nevertheless, there are at least two other simplifications that may be sequentially introduced into the model: the first one is to neglect the vertical dynamics of the wheels, the second one is to neglect the chassis dynamics associated to the suspensions. These models are described more in detail in the next sections.

4.2.2 Massless wheel model (10 dof)

Wheels are responsible of the so-called *hop modes* which have a frequency range of 20−30 Hz. If the wheel mass is neglected in the vertical dynamic equation (4.7), the force exerted by each suspension are exactly balanced by the tyre vertical load. If damping is also neglected, wheel hop equations (4.7) (substituting (4.8), (4.13), (4.10) into it) becomes algebraic and linear in the state variables:

$$K_{tr}^{(r)}(z + \mu b + \phi t_v^{(r)} + z_w^{(rr)}) + K_{ss}^{(r)} z_w^{(rr)} + K_a^{(r)}(z_w^{(rr)} - z_w^{(rl)}) = 0 \tag{4.22a}$$

$$K_{tr}^{(r)}(z + \mu b - \phi t_v^{(r)} + z_w^{(rl)}) + K_{ss}^{(r)} z_w^{(rl)} - K_a^{(r)}(z_w^{(rr)} - z_w^{(rl)}) = 0 \tag{4.22b}$$

$$K_{tr}^{(f)}(z - \mu a + \phi t_v^{(f)} + z_w^{(fr)}) + K_{ss}^{(f)} z_w^{(fr)} + K_a^{(r)}(z_w^{(fr)} - z_w^{(fl)}) = 0 \tag{4.22c}$$

$$K_{tr}^{(f)}(z - \mu a - \phi t_v^{(f)} + z_w^{(fl)}) + K_{ss}^{(f)} z_w^{(fl)} - K_a^{(r)}(z_w^{(fr)} - z_w^{(fl)}) = 0 \tag{4.22d}$$

Therefore the above equations can be solved to express the suspension travel as a function of the other state variables, which can then substituted into (4.10) to find the resulting tyre deformations:

$$\begin{aligned}
\hat{\xi}^{(rr)} &= K_{es}^{(r)}(\mu b + \phi t_v^{(r)} + z) + K_{ea}^{(r)} t_v^{(r)} \phi \\
\hat{\xi}^{(rl)} &= K_{es}^{(r)}(\mu b - \phi t_v^{(r)} + z) - K_{ea}^{(r)} t_v^{(r)} \phi \\
\hat{\xi}^{(fr)} &= K_{es}^{(f)}(-\mu a + \phi t_v^{(f)} + z) + K_{ea}^{(f)} \phi t_v^{(f)} \\
\hat{\xi}^{(fl)} &= K_{es}^{(f)}(-\mu a - \phi t_v^{(f)} + z) - K_{ea}^{(f)} \phi t_v^{(f)}
\end{aligned} \tag{4.23}$$

where

$$\begin{aligned}
K_{es}^{(r)} &= \frac{K_{ss}^{(r)}}{K_{ss}^{(r)} + K_{tr}^{(r)}} & K_{ea}^{(r)} &= \frac{2K_{tr}^{(r)} K_a^{(r)}}{(K_{ss}^{(r)} + K_{tr}^{(r)})(2K_a^{(r)} + K_{ss}^{(r)} + K_{tr}^{(r)})} \\
K_{es}^{(f)} &= \frac{K_{ss}^{(f)}}{K_{ss}^{(f)} + K_{tr}^{(f)}} & K_{ea}^{(f)} &= \frac{2K_a^{(f)} K_{tr}^{(f)}}{(K_{ss}^{(f)} + K_{tr}^{(f)})(2K_a^{(f)} + K_{ss}^{(f)} + K_{tr}^{(f)})}
\end{aligned} \tag{4.24}$$

From the above expressions it can be noticed that tyre deformations are given by two terms: the first is that of a totally rigid suspension with a tyre radial stiffness given by the series of the suspension and tyre springs, and the second term is due to the anti-roll bar and it is proportional to the stiffness of the bars. In particular it can be noticed that $\xi^{(\hat{a}\hat{s})}$ are linear expressions of z, ϕ, μ. To preserve the damping characteristics of the pitch, bounce and roll chassis modes, the damping coefficient of the tyres is then replaced by a modified coefficient K_{ed}, that is an approximated equivalent damping, which preserves the damping ratio of the chassis vibrations modes. Thus, tyre loads are finally given by:

$$N^{(\hat{a}\hat{s})} = f^{+}\left(N_0^{(\hat{a}\hat{s})} + K_{es}^{(\hat{a})} \hat{\xi}^{(\hat{a}\hat{s})} + K_{ed}^{(\hat{a})} \dot{\hat{\xi}}^{(\hat{a}\hat{s})}\right) \tag{4.25}$$

After the elimination of wheel hop equations and suspension travel variables, the first order formulation (4.21) of the ten dof model has 21 state variables:

$$\boldsymbol{x}_{10} = \Big[V;\ \lambda;\ \Omega;\ z;\ \phi;\ \mu;\ z_{dot};\ \phi_{dot};\ \mu_{dot};\ \omega_w^{(\hat{a}\hat{s})};\ F_{ty}^{(\hat{a}\hat{s})};\ F_{tx}^{(\hat{a}\hat{s})}\Big] \tag{4.26}$$

4.2.3 Basic car model (seven dof)

This model neglects the chassis motion due to the suspensions, i.e. variables z, ϕ, μ are set to zero and equations (4.4c), (4.5a), (4.5c) are discarded. Therefore only seven dof there remain: the gross motion variables V, λ, Ω and wheels spin variables $\omega_w^{(\hat{a}\hat{s})}$. However, the introduction of such simplifications determines the loss of information (i.e. the equations) necessary to calculate tyre loads $N^{(\hat{a}\hat{s})}$. To solve this problem, tyre loads are pre-calculated in steady state conditions as a function of the longitudinal speed V, longitudinal acceleration a_x and lateral acceleration a_y using the 14 dof model:

$$N^{(\hat{a}\hat{s})} = N^{(\hat{a}\hat{s})}(V, a_x, a_y) \tag{4.27}$$

More precisely, the full features dynamic model is converted into a steady state problem by equating to zero the right hand side of equations (4.21) (i.e. setting the derivatives to zero), moreover a fictitious longitudinal gravity field has been introduced to emulate longitudinal acceleration a_x, while the steady state lateral acceleration is simply $a_y = \Omega V$. The quasi static problem has 29 algebraic equations and 32 variables: states 4.19, inputs 4.20 and the acceleration a_x. In conclusion, once variables V, a_x, Ω have been fixed, all other variables may be calculated from the steady state equations (provided that a solution exists).

To finally use the quasi-static expression (4.27) in the seven dof dynamic model it is necessary to calculate the longitudinal and lateral accelerations as a function of the state variables. The easiest solution is to add a_x and a_y to the state variable vector, with the following first order low pass filter as corresponding equations:

$$\begin{aligned} \tau_{ax}\dot{a}_x + a_x &= \dot{V} + \Omega V \lambda \\ \tau_{ay}\dot{a}_y + a_y &= \Omega V - \dot{V}\lambda - V\dot{\lambda} \end{aligned} \tag{4.28}$$

where τ_{ax} and τ_{ay} are the filter time constants. Such filters introduce an artificial time lag between tyre loads and vehicle acceleration variations. This can be considered as a simplified description of the time lag which is naturally induced by suspensions in real vehicles.[6]

[6] A possible different approach could consist in modelling the suspension dynamics by using the following differential equation for each tyre load: $\tau_N N' + N = N(a_x, a_y)$, where $N(a_x, a_y)$ is the tyre load in steady state conditions with longitudinal and lateral accelerations a_x, a_y. The time constant τ_N would be such to simulate the correct suspension-induced time lag. This approach however requires two more state variables

In conclusion, the first order formulation (4.21) of the seven dof model has 17 state variables:

$$\boldsymbol{x}_7 = \left[V;\ \lambda;\ \Omega;\ a_x;\ a_y;\ \omega_w^{(\hat{a}\hat{s})};\ F_{ty}^{(\hat{a}\hat{s})};\ F_{tx}^{(\hat{a}\hat{s})}\right] \tag{4.29}$$

4.2.4 Vehicle tracking

The state spaces of the three car models completely determine the car dynamics, however they have to be augmented including the road tracking variables (s, n, α) in order to be usable for minimum lap time problems. When the three variables s, n, α are added to the state variables, three more equations, associated to these variables, must be added to the first order equation system. These three equations can be obtained by expressing the longitudinal and lateral speed of the frame $\boldsymbol{W}_{v0}$ (see (4.3)) both as function of the state variables V, λ, Ω and as function of the time derivatives of the tracking variables $\dot{s}$, $\dot{n}$, $\dot{\alpha}$:

$$V\cos\lambda = \dot{n}\sin\alpha + \dot{s}\cos\alpha(1-\kappa) \tag{4.30a}$$

$$-V\sin\lambda = \dot{n}\cos\alpha - \dot{s}\sin\alpha(1-\kappa) \tag{4.30b}$$

$$\Omega = \dot{\alpha} + \dot{s}\kappa \tag{4.30c}$$

where κ is the road curvature (see appendix A). These equations are the first order equations that determine the time evolution of the tracking variables. They can also be made explicit in $\dot{s}$, $\dot{n}$, $\dot{\alpha}$:

$$\dot{s} = \frac{V\cos(\alpha-\lambda)}{1-n\kappa(s)} \tag{4.31a}$$

$$\dot{n} = V\sin(\alpha-\lambda) \tag{4.31b}$$

$$\dot{\alpha} = \Omega - \kappa(s)\frac{V\cos(\alpha-\lambda)}{1-n\kappa(s)} \tag{4.31c}$$

Since the sideslip angle λ has been assumed small, the simplification $\sin\lambda \approx \lambda$ and $\cos\lambda \approx 1$ can be used.

4.2.5 Formulation of the minimum lap time OCP

The minimum lap time problem are usually described in the space domain rather than in the time domain since practical evidence shows that the former makes the resulting optimal control problem to be easier to solve [25, 59, 57]. Thus, the time to space transformation described in appendix A is used to formulate the OCP in the road curvilinear abscissa domain. The expression for $\dot{s}$ (4.31a) shows that the time to space transformation is not

than the used one.

singular provided that $V > 0$, i.e. the vehicle never stops, and $n < 1/\kappa(s)$, i.e. the vehicle never passes over the local curvature centre of the road.

Optimal control solutions typically include abrupt variation of the control inputs, which conflict with the dislike of jerk felt by human drivers [93, 13, 15]. To obtain smoother manoeuvres, the model is therefore controlled by the steering angle rate and the longitudinal jerk. According to this, the two driver inputs τ_d and δ have been included to the state variables, while the controls have been moved to their time derivatives, j_τ and j_δ, satisfying the following equations:

$$j_\delta = \dot{\delta} \qquad j_\tau = \dot{\tau}_d \tag{4.32}$$

where j_τ is the longitudinal jerk control, which (mainly) controls the longitudinal dynamics, and j_δ is the lateral jerk control, which (mainly) controls the lateral dynamics. Thus, the optimal control state variables include the car model variables (either 14, ten or seven depending on the model), plus the tracking variables n and α, and the driver inputs τ_d and δ. The controls are the derivatives of the driver inputs, j_δ and j_τ. The road curvilinear abscissa is instead the domain variable.

In order to minimise the lap time, the OCP target is composed only of a Lagrange term equal to $1/\dot{s}$, such that its integral along the road curvilinear abscissa is equal to the lap time. Equation (4.31a) clearly shows that the quantity $1/\dot{s}$ can be easily expressed as function of the other state variables.

The OCP boundary conditions are cyclic, meaning that all the state variables at the end of the domain (i.e. when s equals the road length) must be equal to their initial value (i.e. $s = 0$).

A constraint is added to the optimal control problem in order to prevent the vehicle from exiting from the track:

$$-\left(\frac{r_w}{2} - t_v\right) < n < \frac{r_w}{2} - t_v \tag{4.33}$$

where t_v is the maximum between the rear $t_v^{(r)}$ and front $t_v^{(f)}$ half width of the car, and r_w is the road width (that is a function of s). Another constrain is used to limit the overall traction torque τ_d to maximum engine torque available $\tau_e^{(max)}$ as follows:

$$\tau_d < \tau_g \tau_e^{(max)} \left(\tau_g \frac{\omega_w^{(rr)} + \omega_w^{(rl)}}{2}\right) \tag{4.34}$$

where τ_g is the engine to rear axle gear ratio. It should be noted that $\tau_g \tau_e^{(max)}$ is the torque available at the rear axle, assuming that the proper gearbox ratio is used while driving. Moreover, the maximum engine torque depends on the engine spin velocity. A third constraint has been added to the OCP in order to prevent the ideal driver from locking the front wheels when braking: indeed real drivers tend to avoid such manoeuvre as long as it causes high tyre wear and makes the tyre to loose performance. Mathematically this constraint has been expressed as:

$$\kappa^{(fr)} \geq \kappa_{min} \qquad \kappa^{(fl)} \geq \kappa_{min} \tag{4.35}$$

where $\kappa_{min} \approx -0.6$ is the minimum slip value as it would result in unacceptable tyre wear.

Finally, the control inputs j_τ, j_δ, which are strictly related to the driving torque and steering angle rate by equations (4.32), are limited in magnitude so as to avoid control rates higher than what a human driver can sustain:

$$-j_{\tau,max} \leq j_\tau \leq j_{\tau,max} \qquad - j_{\delta,max} \leq j_y \leq j_{\delta,max} \tag{4.36}$$

where $j_{\tau,max}$ and $j_{\delta,max}$ are the maximum value allowed for j_τ and j_δ controls.

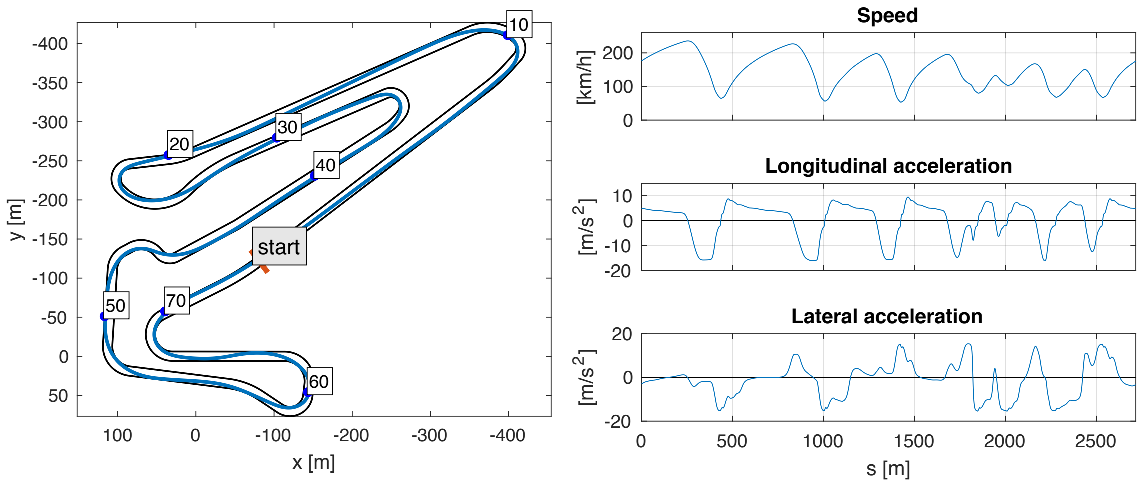
The resulting optimal control problem is solved using *Pins*, the numeric OCP solver based on an indirect approach (see section 3.2.1).

4.2.6 Simulations results

The three models described in this section have been compared on a full lap simulation on the Adria International Raceway. The trajectory resulting from the 14 dof model simulation is illustrated in figure 4.2a, while the vehicle speed and accelerations are shown in figure 4.2b. The maximum braking and lateral accelerations are $\approx 15m/s^2$, and the maximum positive longitudinal one is $\approx 10m/s^2$, limited by the engine power limit. The simulated lap time is of $73.789s$, which is not so different from a qualifying lap-time of the GT1 series.

The computational time for a single simulation is respectively $\approx 1700s$ on a standard PC equipped with an *Intel Core 2 Quad* Q9300 CPU. Simulations have been repeated for the ten dof and seven dof models, finding simulated lap times respectively of $73.790s$ and $72.967s$, while the computational time was respectively $970s$ and $560s$. In other words, simplified models lead to a lap time prediction very similar to the one of the 14 dof model, but using respectively only the 57% and 33% of computational time.

In lap time simulations it is more important to capture the lap time sensitivity to vehicle setup variations that its absolute value, which is quite difficult to find because of the uncertainty of some parameters, first of all tyre adherence in real conditions. Thus the accuracy of the three models is tested through a comparison in the sensitivity with respect to the variation of two car characteristics, rather than looking at the absolute lap-time for each



(a) Trajectory

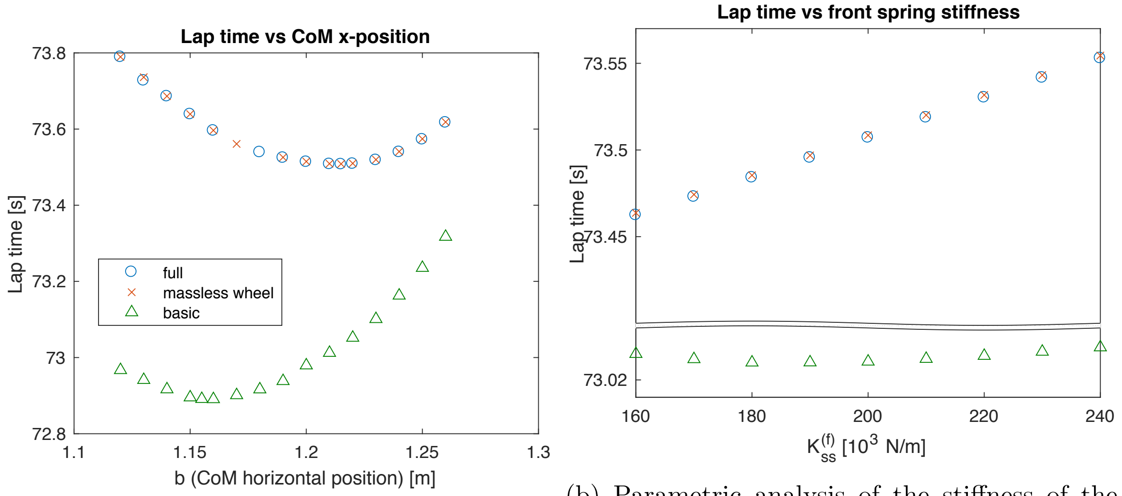
(b) Gross-motion

Figure 4.2: Trajectory and gross-motion resulting from optimal control simulation on the circuit of Adria. The track is treaded counter-clockwise.

model. The first parameter that has been analysed is the longitudinal position of the centre of mass: its distance from the rear axle has been varied in the range $1.12m < b < 1.26m$. The lap-times obtained with the three models are shown in figure 4.3a. The 14 dof model shows a kind of parabolic trend, with the best performance obtained for $b = 1.215m$. The ten dof model lap-times are very close to that of the full one: the highest time difference is of $\approx 0.01s$ and the minimum time also is reached at the same value of b. Differently, the basic seven dof model shows some discrepancies with respect to the other two models: the lap-times are significantly lower and, more important, the minimum time is obtained at a different position of the CoM, precisely for $b = 1.155m$.

In the second parametric analysis, the stiffness of the front suspensions spring has been varied in the range $160 \times 10^3 N/m < K_{ss}^{(f)} < 240 \times 10^3 N/m$, and the calculated lap-times are shown in figure 4.3b. Again, the lap-times obtained with the full and massless-wheel models are very close, and the minimum time is obtained for the lowest value of the spring stiffness (while the absolute minimum seems to be located outside interval of analysis). In the range of stiffness considered, the lap-times varies of $\approx 0.1s$ between the lowest and the highest. On the contrary, the basic model appears to be less sensitive to changes of the suspension stiffness, with lap-times varying of only $0.01s$ in the same range of stiffness. Moreover, a local minimum lap time is shown for $K_{ss}^{(f)} = 180 \times 10^3 N/m$.

Since the main difference between the three models lays in the way tyres loads are calculated, the results of the previous analyses suggest that the steady state tyre loads approach adopted for the basic model does not manage to faithfully reproduce the tyres



(a) Parametric analysis of the x-position of the CoM b (the reference value is $b = 1.120m$).

(b) Parametric analysis of the stiffness of the front suspensions spring $K_{ss}^{(f)}$ (reference is $K_{ss}^{(f)} = 200 \times 10^3 N/m$).

Figure 4.3: Parametric analyses comparison of the three models.

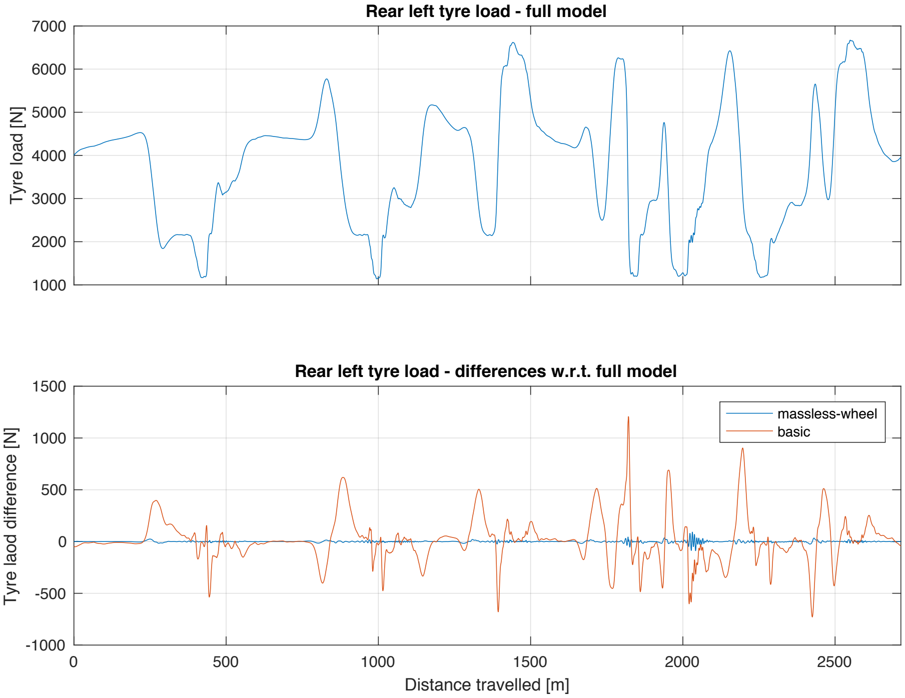


Figure 4.4: Rear left tyre load for the full model (top plot) and difference with respect the other two models (bottom plot). While for the massless wheel model tyre load differences are of few tens of Newton, for the basic model differences arrive up to $1kN$. In the simulation for the basic model the trajectory of the full model has been imposed in order to remove trajectory-related load differences.

load dynamics. In order to confirm this, the simple model has been compared on the exact same trajectory of the full model, so as to remove any trajectory-related load differences. Thus, a simulation has been performed with the basic model on an imposed trajectory (the one obtained with the full model). The load on the rear left tyre resulting from this simulation is shown in figure 4.4. In the top plot the rear left tyre load for the full model is presented; in the bottom plot the differences with respect the other two models are highlighted. The basic model shows a tyre load difference of up to $1 \times 10^3 N$ with respect to the full model, while the massless-wheel one differs up to only $80N$.

The analysis of tyre load transfer, together with the results of the parametric analyses, suggests that the the massless-wheel model represents a useful simplification with respect to the full one, since it reduces the simulation computing time and at the same time it preserver a good accuracy on the simulated vehicle dynamics (in particular of tyre loads). On the contrary, the basic seven dof model leads to different simulation outcomes in both the parametric analysis, thus the model results to be over-simplified. It might be possible that the differences between the three dynamics modelling levels can increase or diminish if the dataset of a different car category are used. However, it has been shown that there are differences between the models and also a rough quantification of them has been provided.

4.3 GP2 race car

In this section a novel and accurate GP2 car model for optimal control problems is developed and used for OC lap time simulations. In the previous section 4.2 it has been shown that car models with quasi steady state tyre loads may lead to significantly different design optimisation outcomes and therefore more accurate car models should be used in lap time simulations, at least when tyre loads dynamics is of relevant importance. Tyre load transfer can be dynamically simulated by including in the car model all the six chassis dof together with the four suspension-related ones. Compared to a QSS tyre loads model, this approach allows to dynamically simulate load transfers taking into account for suspension effects, at the expense of three or seven additional dof, as seen in section 4.2. In the same section, it has been shown that suspension degrees of freedom may be neglected if suspension travels can be expressed (or estimated) from the other state variables. The GP2 model here presented extends the models of the previous section by taking into consideration for the suspension kinematics and for a more complex aerodynamic forces. The increase of the model complexity is such that it is no longer possible to express the suspension travel as function of the other state variables, as in the massless wheel model. Indeed, the nonlinearities of the suspension kinematics and aerodynamic forces make the suspension force balance equation not analytically solvable.

In the next subsection the multibody model is described and the corresponding equations of motion are presented. The multibody model is composed by the main chassis (6 dof) and four wheels, which are connected to the chassis by means of the suspension system; wheels can spin about their spin axis (4 dof) and move with the suspension travel (other 4 dof). Suspension kinematics is accurately analysed in order to understand how tyre lateral and longitudinal forces contribute to the vehicle trim and thus load transfer. Then, the minimum lap time simulation outcomes are presented and the results are validated through comparison with experimental data acquired during a qualifying lap on the Montmelò circuit in 2012.

4.3.1 Multibody model of the GP2 Car

A GP2 car is a rear wheel drive formula car characterized by very stiff suspensions and high aerodynamic downforce generated by rear and front wings. The aerodynamics of GP2 cars is quite complex and wing force intensity depends on ride heights, therefore any model used to simulate GP2 car dynamics should carefully reproduce not only aerodynamic forces, but also all parameters that determine vehicle trim, first of all suspensions and tyres. The model here developed abandons the quasi-steady state tyre load simplification since it has been showed that it negatively influences simulations outcomes (as discussed in section 4.2). Suspension kinematics are taken into consideration because they have a relevant influence on vehicle performance [6, 64]. A car model that includes this features leads to a relatively complex minimum lap time optimal control problem, that may be very difficult to solve. The car mathematical model is thus a key element, together with the optimal control formulation and software used, to successfully solve the resulting problem.

The chassis is modelled exactly as it has been done in section 4.2.1 for the 14 dof car model. The chassis is a rigid body that can translate and rotate about the three axes (SAE axis convention is adopted); it is recalled that position variables associated to these six degrees of freedom (dof) are: the position along the curvilinear abscissa s, the lateral displacement from the road middle lane n, the CoM vertical displacement z w.r.t its nominal height h, the chassis roll ϕ and pitch μ angles, and the chassis yaw angle relative to the road middle lane direction α. Wheels are connected to the main chassis through short-long arm suspensions, that introduce one additional dof per wheel; the vertical (i.e. along the z axis of the chassis frame) motion of a generic wheel is named $z_w^{(\hat{a}\hat{s})}$[7]. As the suspension moves, the actual rigid motion of the wheel plate has been included in the model, including translations and rotations about all axes. Suspension forces take into account coil springs, torsion bars, dampers and anti-roll bars. Finally, wheel spins add four more dof $\omega_w^{(\hat{a}\hat{s})}$. All the degrees of freedom are the same of the 14 dof car model, reported in table 4.1. A speed sensitive differential is present at the rear axle. Tyre longitudinal forces are calculated through a nonlinear tyre model, and aerodynamic drag and lift forces depend on ride heights. Summarizing, the model comprises the same 14 mechanical dof as the car model presented in section 4.2.1; the list of the 14 dof can be found in table 4.1. Additional variables are listed in table 4.4.

Table 4.4: GP2 model variables.

variable	units	description
V	m/s	chassis on-road speed
λ	rad	chassis drift angle
Ω	rad/s	chassis yaw rate
y_{sr}	m	steering rack displacement
$h^{(f)}$	m	front ride height
$h^{(r)}$	m	rear ride height

[7] It is recalled that the superscript $\hat{a} \in \{r, f\}$ indicates the rear or front axle and $\hat{s} \in \{r, l\}$ indicates the right or left side.

variable	units	description
κ	m^{-1}	road curvature
$\delta^{(\hat{a}\hat{s})}$	rad	wheel steering angle
$\phi_w^{(\hat{a}\hat{s})}$	rad	wheel camber angle w.r.t the road plane
$x_w^{(\hat{a}\hat{s})}$	m	wheel plate displacement along x-axis (with respect to nominal position)
$y_w^{(\hat{a}\hat{s})}$	m	wheel plate displacement along y-axis (with respect to nominal position)
$\gamma_w^{(\hat{a}\hat{s})}$	rad	wheel plate camber angle (w.r.t the nominal position)
$\mu_w^{(\hat{a}\hat{s})}$	rad	wheel plate pitch angle (w.r.t the nominal position)
$\psi^{(\hat{a}\hat{s})}$	rad	wheel plate yaw (steering) angle (w.r.t the nominal position)
$N^{(\hat{a}\hat{s})}$	N	tyre load
$F_{tx}^{(\hat{a}\hat{s})}$	N	tyre longitudinal force
$F_{ty}^{(\hat{a}\hat{s})}$	N	tyre lateral force
τ_e	Nm	engine torque
τ_b	Nm	braking torque
$\tau_{wd}^{(\hat{a}\hat{s})}$	Nm	driving torque applied to the wheel
$F_s^{(\hat{a}\hat{s})}$	N	suspension force acting on the chassis and counter-reacting on the wheel
F_d	N	drag force
$F_l^{(f)}$	N	front axle aero-downforce
$F_l^{(r)}$	N	rear axle aero-downforce
R_t	N	tyre rolling resistance
$\xi^{(\hat{a}\hat{s})}$	m	tyre radial deformation
$\kappa_t^{(\hat{a}\hat{s})}$	-	tyre longitudinal slip
$\lambda_t^{(\hat{a}\hat{s})}$	rad	tyre sideslip angle
$V_{tx}^{(\hat{a}\hat{s})}$	m/s	tyre contact point longitudinal speed
$V_{ts}^{(\hat{a}\hat{s})}$	m/s	tyre contact point spinning speed
$V_{ty}^{(\hat{a}\hat{s})}$	m/s	tyre contact point lateral speed

4.3.1.1 Suspension kinematics

In a GP2 car the suspension system is based on short-long arm (SLA) type, as shown in figure 4.5: the wheel is connected to the chassis by means of two A-shaped arms that end with spherical joints, plus the steering rod which ends either in a movable spherical joint, in the case of the front suspension, or in a fixed spherical joint, in the case of the rear one (even if the rear wheels have no steer, this latter rod will be referred to as "steering rod", as for the front suspension). The position of attachment points together with rod lengths completely determine the rigid motion that the wheel follows when the suspension moves up and down. The knowledge of the precise suspension kinematics is important to accurately reproduce the load transfers, even in stationary conditions. Indeed, since the tyre contact point moves fore and aft as well as left and right in the road plane as the suspension moves up and down, tyre lateral and longitudinal forces influence the equilibrium of the suspension. Moreover, each wheel not only translates as the suspension moves, but it also rotates about all the three axes; such rotations influence the in-plane forces generated by the tyre as long as they modify both the tyre sideslip and camber angles.

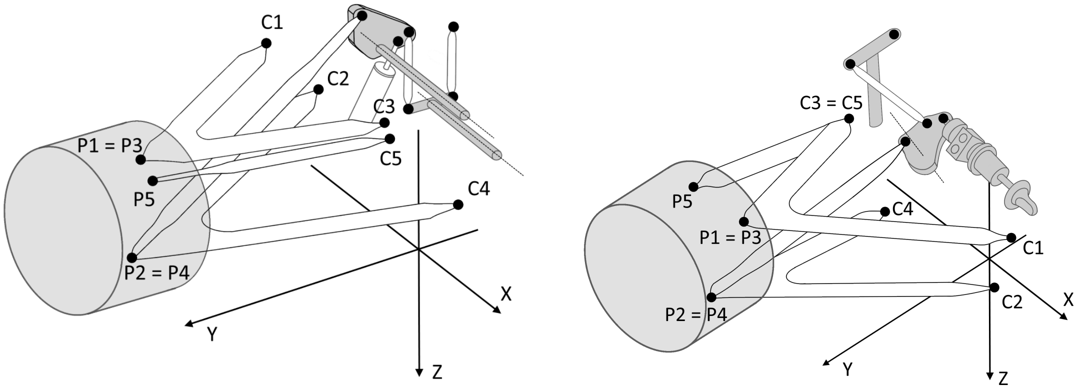


Figure 4.5: Front (left) and rear (right) short-long arm suspension scheme.

Here the kinematics of the 2011 Dallara GP2 car will be analysed, while some numerical results for the steady state configuration will be provided in section 4.3.1.4. From the kinematic point of view, the SLA suspension is a special case of the multilink suspension and it is composed by five rods attached through spherical joints to the chassis at one edge, and to the wheel plate at the other edge, as shown in figure 4.5. Attachments points that are fixed to the vehicle frame and are named: upper front chassis point $\mathbf{C}^1$, lower front chassis point $\mathbf{C}^2$, upper rear chassis point $\mathbf{C}^3$, lower rear chassis point $\mathbf{C}^4$ and steer chassis point $\mathbf{C}^5$. This latter point, which is connected to the steering rod, is fixed to the chassis in the case of the rear suspension, while in the front one it moves along the y-axis as the driver steers. On the wheel side, attachments points are named: upper front wheel point $\mathbf{P}^1$, lower front wheel point $\mathbf{P}^2$, upper rear wheel point $\mathbf{P}^3$, lower rear wheel point $\mathbf{P}^4$ and steer wheel point $\mathbf{P}^5$. Due to the design of this SLA suspensions, the point $\mathbf{P}^1$ coincide with $\mathbf{P}^3$, and $\mathbf{P}^2$ with $\mathbf{P}^4$ (see figure 4.5). The wheel plate position and orientation can be completely described by the displacement of the wheel plate centre w.r.t its nominal configuration $(x_w^{(\hat{a}\hat{s})}, y_w^{(\hat{a}\hat{s})}, z_w^{(\hat{a}\hat{s})})$, the yaw angle $\psi^{(\hat{a}\hat{s})}$, the camber angle $\gamma_w^{(\hat{a}\hat{s})}$ and the spin angle $\mu_w^{(\hat{a}\hat{s})}$:

$$\boldsymbol{w}^{(\hat{a}\hat{s})} = \left[x_w^{(\hat{a}\hat{s})}; y_w^{(\hat{a}\hat{s})}; z_w^{(\hat{a}\hat{s})}; \gamma_w^{(\hat{a}\hat{s})}; \mu_w^{(\hat{a}\hat{s})}; \psi^{(\hat{a}\hat{s})}\right] \tag{4.37}$$

The rigid motion of the wheel is described by the 4 × 4 transformation matrix method as follows:

$$\boldsymbol{W}(\boldsymbol{w}^{(\hat{a}\hat{s})}) = \boldsymbol{W}_0^{(\hat{a}\hat{s})}\boldsymbol{T}(x_w^{(\hat{a}\hat{s})}, y_w^{(\hat{a}\hat{s})}, z_w^{(\hat{a}\hat{s})})\boldsymbol{R}_z(\psi^{(\hat{a}\hat{s})})\boldsymbol{R}_x(\gamma_w^{(\hat{a}\hat{s})})\boldsymbol{R}_y(\mu_w^{(\hat{a}\hat{s})}) \equiv \boldsymbol{W}_0^{(\hat{a}\hat{s})}\boldsymbol{S}(\boldsymbol{w}^{(\hat{a}\hat{s})}) \tag{4.38}$$

where $\boldsymbol{W}_0^{(\hat{a}\hat{s})}$ is the wheel reference system in nominal conditions, $\boldsymbol{T}(x_w^{(\hat{a}\hat{s})}, y_w^{(\hat{a}\hat{s})}, x_w^{(\hat{a}\hat{s})})$ is the translation transformation matrix, and $\boldsymbol{R}_i(a)$ are the rotation matrices around i-axis of an angle a. From the equivalence in (4.38), $\boldsymbol{S}(\boldsymbol{w}^{(\hat{a}\hat{s})})$ is the transformation matrix that gives the wheel plate configuration with respect to its nominal position. Since the suspension linkages allow the wheel to have only one degree of freedom, the coordinates $\boldsymbol{w}$ are mutually dependent and can be expressed as function of only one independent parameter; the wheel plate vertical displacement $z_w^{(\hat{a}\hat{s})}$ has been chosen as independent dof since it represents the most important movement of the suspension. The five mathematical constraints necessary to remove the dependent variables can be obtained by imposing that the distance between the connecting points located at the extremities of each rod must be equal to the rod length. As the wheel plate moves, the coordinates of each connecting point on the wheel side $\boldsymbol{P}^i = (x_{P^i}, y_{P^i}, z_{P^i}, 1)$ can be easily calculated from their nominal position $\boldsymbol{P}_0^i$:

$$\boldsymbol{P}^{i(\hat{a}\hat{s})} = \boldsymbol{W}_0^{(\hat{a}\hat{s})}\boldsymbol{S}(\boldsymbol{w}^{(\hat{a}\hat{s})})\boldsymbol{W}_0^{-1(\hat{a}\hat{s})}\boldsymbol{P}_0^{i(\hat{a}\hat{s})} \tag{4.39}$$

Therefore the five constraints (for each wheel) can be expressed by the following relationship:

$$(x_{P^i} - x_{C^i})^2 + (y_{P^i} - y_{C^i})^2 + (z_{P^i} - z_{C^i})^2 - l_i^2 = 0, \qquad i \in \{1..5\}, \tag{4.40}$$

where l_i is the length of the rod connecting $\boldsymbol{P}^i$ with $\boldsymbol{C}^i$. In conclusion, a set of five algebraic constraint equations is obtained for each suspension:

$$\phi_i^{(\hat{a}\hat{s})}\left(\boldsymbol{w}^{(\hat{a}\hat{s})}\right) = 0, \qquad i \in 1..5 \tag{4.41}$$

These constraint equations have been solved numerically for the given suspension geometry, both rear and front. The displacement of the wheel centre $x_w^{(rr)}, y_w^{(rr)}$, as well as camber $\gamma_w^{(rr)}$, pitch $\mu_w^{(rr)}$ and steer $\psi_w^{(rr)}$ angles of the rear right wheel plate are depicted as a function of the vertical travel $z_w^{(rr)}$ in figure 4.6. The figure shows that the wheel plate translation along x and y direction are less than $1mm$, moreover the camber and steer angles are smaller than 0.5° and 0.02°. The trend of x, y, γ, μ and δ as function of z has been fitted by polynomials up to the second order, and the resulting fittings are shown in figure 4.6 by continuous lines. The fitting of such variables is required to analytically express all the movements of the wheel plate as function of z, as it is necessary for the development of the multibody model described in the next section. In the front suspension the position of the steering chassis point ($\mathbf{P}^5$) along the y-axis is controlled by the driver input on the steering wheel, therefore the movements of the front wheel plate ($x_w^{(f\hat{s})}, y_w^{(f\hat{s})}, \gamma_w^{(f\hat{s})}, \mu_w^{(f\hat{s})}, \psi_w^{(f\hat{s})}$) depend both on the wheel vertical shift $z_w^{(f\hat{s})}$ and on the steering rack lateral displacement y_{sr}. Figure 4.7 shows the front right wheel motion as function of $z_w^{(fr)}$ when the steering is null, while figure 4.8 shows the same quantities as function of y_{sr} when the suspension travel is zero. Similarly to the rear suspension, the variables $x_w^{(fr)}, y_w^{(fr)}, \gamma_w^{(fr)}, \mu_w^{(fr)}, \psi_w^{(fr)}$ have been fitted by polynomials up to the second order as function of $z_w^{(fr)}$ and y_{sr}, and the resulting fits are shown in the figures by continuous lines.

Figures 4.6 to 4.8 demonstrate that the rotation angles of both rear and front wheel plate are small; indeed the largest angle is the yaw one of the front wheels, that is typically limited to ±10°, while the other angles are lower than 0.5°. Since such angles are small, they can be linearised, i.e. the rotations of such angles can be expanded in Taylor series up to the first order. The transformation matrix (4.38) then becomes:

$$\boldsymbol{W}(\boldsymbol{w}^{(\hat{a}\hat{s})}) = \boldsymbol{W}_0^{(\hat{a}\hat{s})} \begin{bmatrix} 1 & -\psi_w^{(\hat{a}\hat{s})} & \mu_w^{(\hat{a}\hat{s})} & x_w^{(\hat{a}\hat{s})} \\ \psi_w^{(\hat{a}\hat{s})} & 1 & -\gamma_w^{(\hat{a}\hat{s})} & y_w^{(\hat{a}\hat{s})} \\ -\mu_w^{(\hat{a}\hat{s})} & \gamma_w^{(\hat{a}\hat{s})} & 1 & z_w^{(\hat{a}\hat{s})} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.42}$$

Even if $x_w^{(\hat{a}\hat{s})}, y_w^{(\hat{a}\hat{s})}, \gamma_w^{(\hat{a}\hat{s})}, \psi^{(\hat{a}\hat{s})}$ are infinitesimal, they have a noticeable effect on the tyre forces: indeed a small camber angle of $\approx 0.2°$ generates a lateral force variation of $2-3\%$. The study of which terms are significant for the overall car performance and which are negligible will be presented in section 4.3.1.4. The transformation matrix (4.42), where the

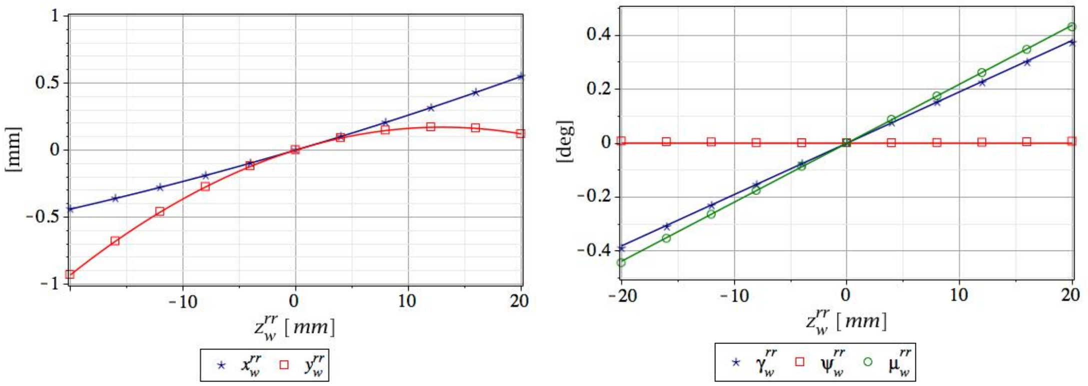


Figure 4.6: Rear wheel plate movements: $x_w^{(rr)}, y_w^{(rr)}$ displacements and $\gamma_w^{(rr)}$, $\psi_w^{(rr)}$, $\mu_w^{(rr)}$ angles are represented as function of the wheel vertical travel $z_w^{(rr)}$. The continuous lines represent the fitting.

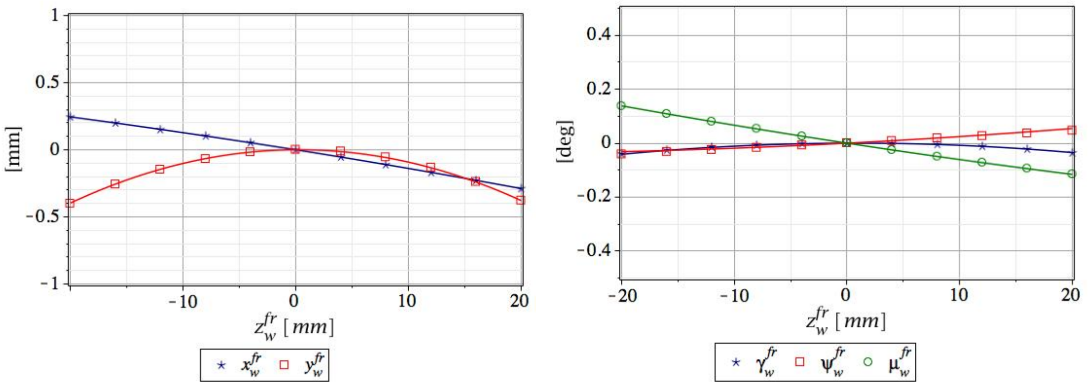


Figure 4.7: Front wheel plate movements: $x_w^{(fr)}, y_w^{(fr)}$ displacements and $\gamma_w^{(fr)}$, $\psi_w^{(fr)}$, $\mu_w^{(fr)}$ angles are represented as function of the wheel vertical travel $z_w^{(fr)}$ when $y_{sr} = 0$. The continuous lines represent the fitting.

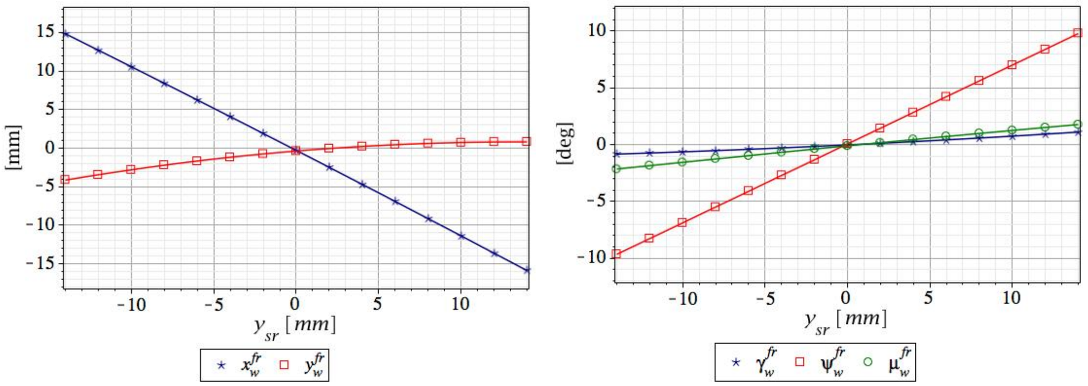


Figure 4.8: Front wheel plate movements: $x_w^{(fr)}, y_w^{(fr)}$ displacements and $\gamma_w^{(fr)}$, $\psi_w^{(fr)}$, $\mu_w^{(fr)}$ angles are represented as function of the steering rack displacement y_{sr} when the vertical displacement is null $z_w^{(fr)} = 0$. The continuous lines represent the fitting.

Table 4.5: Spring and damper travels.

variable and fit	description
$\upsilon^{(r\hat{s})} = -0.832 z_w^{(r\hat{s})}$	rear dampers and coil springs travel
$\upsilon^{(fr)} = -0.879 z_w^{(fr)} + 0.0662 y_{sr}$	front right damper travel
$\upsilon^{(fl)} = -0.879 z_w^{(fl)} - 0.0662 y_{sr}$	front left damper travel
$\theta_{fr} = -0.722 z_w^{(fr)} + 0.0541 y_{sr}$	front right torsion bar angle[1]
$\theta_{fl} = -0.722 z_w^{(fl)} - 0.0541 y_{sr}$	front left torsion bar angle[1]

[1]: the coefficients are expressed in degrees per millimetre
Ranges are: $-20\,\mathrm{mm} < z_w^{(\hat{a}\hat{s})} < 20\,\mathrm{mm}$, $-14\,\mathrm{mm} < y_{sr} < 14\,\mathrm{mm}$

quantities $x_w^{(\hat{a}\hat{s})}, y_w^{(\hat{a}\hat{s})}, \gamma_w^{(\hat{a}\hat{s})}, \mu_w^{(\hat{a}\hat{s})}, \psi^{(\hat{a}\hat{s})}$ are replaced by 2^{nd} degree polynomials in $z_w^{(\hat{a}\hat{s})}$ (and y_{sr}), will be used in the development of the multibody car model described in the next section.

The analysis of suspension kinematics is also necessary to calculate the equivalent vertical force at wheel of the torsion bars, coil springs, dampers and anti-roll bars. The rear wheels are connected through a push-rod and a rocker to the dampers and the coil springs, as shown in figure 4.5. The rear anti-roll bar is connected to the left and right rocker through two link-bars. In the front suspensions, again, a push-rod connects the wheel to a rocker; however, this rocker is not free to rotate around its pivot because it is connected to the chassis with a torsion bar. Thus, when the suspension moves, the elastic force is exerted by this torsion bar. Then, the damper and the anti-roll bar are attached to the rocker similarly to the rear suspension design. The travel of the dampers, rear coil springs and front torsion bars has again been fitted by polynomials as functions of $z_w^{(\hat{a}\hat{s})}$ and y_{sr}, and the resulting polynomial coefficients are reported in table 4.5

4.3.1.2 Equations of motion

In this section the equations of motion for the GP2 car model are calculated. The same road and vehicle tracking model described in section 4.2.1 is used here. The $2D$ road model is used since the Montmelò circuit (that is where the lap time simulations will be performed) does not present relevant elevation variations. As in section 4.2.1 the relative frame $\boldsymbol{W}_{rel}$ that describes the motion of the chassis w.r.t the road middle lane position (see equation (A.6)) is given by the following transformation sequence: a lateral translation of n, a rotation about the z axis of α, a vertical displacement of $z - h$ (h is the nominal CoM height from ground), a roll rotation of ϕ and a pitch rotation of μ. In a GP2 car, since the suspensions are very stiff (significantly more than in GT cars), the chassis vertical displacement z, roll ϕ and pitch angles μ are very small, indeed they lay in the ranges

$0\,\mathrm{mm} < z < 25\,\mathrm{mm}$, $|\phi| < 1°$, $|\mu| < 0.5°$). Thus the resulting chassis relative reference system $\boldsymbol{W}_{rel}$ therefore is the same reported in equation (4.2).

Newton and Euler equations of motion are still calculated within the frame $\boldsymbol{W}_{v0}$ defined in equation (4.3). In the Newton and Euler equations for the full vehicle the acceleration terms related to the second order derivative of the wheel plate longitudinal $x_w^{(\hat{a}\hat{s})}$ and lateral $y_w^{(\hat{a}\hat{s})}$ displacements, as well as of the wheel plate camber $\phi_w^{(\hat{a}\hat{s})}$, yaw $\psi^{(\hat{a}\hat{s})}$ and spin $\mu_w^{(\hat{a}\hat{s})}$ angles, have been neglected, since they are small compared to the acceleration terms related to the chassis (z, ϕ, μ) and wheel plate vertical displacement $z_w^{(\hat{a}\hat{s})}$ (see section 4.3.1.1). Moreover, the drift and wheel steering angles have been considered small since it is always lower than ≈ 10° in the case of a GP2 car. Newton and Euler equations are therefore almost the same of the one described for the 14 dof GT car model (4.4) (4.5), yet some differences are present. Indeed, in the GP2 model the tyre rolling resistance is considered, the aerodynamic downforce is divided into the front and rear axle components, and all the wheels have a steering angle. Thus, Newton equations are:

$$m(\Omega V\lambda + \dot{V}) = \sum_{\hat{a}\hat{s}} F_{tx}^{(\hat{a}\hat{s})} - \sum_{\hat{a}\hat{s}} \delta_w^{(\hat{a}\hat{s})} F_{tx}^{(\hat{a}\hat{s})} - F_d - R_t \tag{4.43a}$$

$$m(\Omega V - \dot{V}\lambda - V\dot{\lambda}) = +\sum_{\hat{a}\hat{s}} \delta_w^{(\hat{a}\hat{s})} F_{tx}^{(\hat{a}\hat{s})} \tag{4.43b}$$

$$m(\ddot{z} - g) + \sum_{\hat{a}\hat{s}} m_w^{(\hat{a})} \ddot{z}_w^{(\hat{a}\hat{s})} = -F_l^{(f)} - F_l^{(r)} - \sum_{\hat{a}\hat{s}} N^{(\hat{a}\hat{s})} \tag{4.43c}$$

where V is the vehicle speed, λ the sideslip angle, Ω is the car yaw rate (about the axis perpendicular to the road plane), $F_{tx}^{(\hat{a}\hat{s})}$ and $F_{ty}^{(\hat{a}\hat{s})}$ are the longitudinal and lateral tyre forces, $m_w^{(\hat{a})}$ are wheel masses, F_d is the drag force, $F_l^{(f)}$, $F_l^{(r)}$ are the aerodynamic downforce at the front and rear axles and R_t is the rolling resistance of the four wheels.

Under the same assumptions used for the Newton equations, the Euler equations are almost the same of the 14 dof GT car model (4.5), with the differences that the steering angles of the rear wheels now appear, moreover the x, y and z component of the net external torque acting on the chassis τ_x, τ_y, and τ_z are different due to the displacements of the wheel plates. Thus the updated Euler equations are (still calculated w.r.t the origin of the frame

$\boldsymbol{W}_{v0}$ defined in (4.3)):

$$\begin{aligned}
I_{xx}\ddot{\phi} + \left(I_{zz}\mu - I_{xx}\mu - I_{xz} - \sum_{\hat{a}\hat{s}} m_w^{(\hat{a})} b^{(\hat{a})} z_w^{(\hat{a}\hat{s})}\right)\dot{\Omega}& \\
+(I_{zz} - I_{xx} - I_{yy})\Omega\dot{\mu} - M\left(hV\dot{\lambda} + h\lambda\dot{V} + (z-h)\Omega V\right)& \\
-\sum_{\hat{a}\hat{s}} m_w^{(\hat{a})} t_v^{(\hat{a}\hat{s})} \ddot{z}_w^{(\hat{a}\hat{s})} - \left((I_{yy} - I_{zz})\phi + \sum_{\hat{a}\hat{s}} m_w^{(\hat{a})} t_v^{(\hat{a}\hat{s})} z_w^{(\hat{a}\hat{s})}\right)\Omega^2& \\
-\sum_{\hat{a}\hat{s}} m_w^{(\hat{a})} z_w^{(\hat{a}\hat{s})}\Omega V + I_w^{(f)}(\dot{\omega}_w^{(fr)} + \dot{\omega}_w^{(fl)})\delta& \\
+I_w^{(f)}(\omega_w^{(fr)} + \omega_w^{(fl)})\dot{\delta} + \sum_{\hat{a}\hat{s}} I_w^{(\hat{a})}\omega_w^{(\hat{a}\hat{s})}\Omega = \tau_x&
\end{aligned} \tag{4.44a}$$

$$\begin{aligned}
I_{yy}\ddot{\mu} + \left((I_{yy} - I_{zz})\phi + \sum_{\hat{a}\hat{s}} m_w^{(\hat{a})} t_v^{(\hat{a}\hat{s})} z_w^{(\hat{a}\hat{s})}\right)\dot{\Omega} + (I_{xx} + I_{yy} - I_{zz})\Omega\dot{\phi}& \\
+M\left((z-h)\dot{V} - h\lambda\Omega V\right) + \sum_{\hat{a}\hat{s}} m_w^{(\hat{a})} b^{(\hat{a})} z_w^{(\hat{a}\hat{s})} + \sum_{\hat{a}\hat{s}} m_w^{(\hat{a})} z_w^{(\hat{a}\hat{s})}\dot{V}& \\
-\sum_{\hat{a}\hat{s}} I_w^{(\hat{a}\hat{s})}\dot{\omega}_w^{(\hat{a}\hat{s})} + \left((I_{zz} - I_{xx})\mu - I_{xz} + \sum_{\hat{a}\hat{s}} m_w^{(\hat{a})} b^{(\hat{a})} z_w^{(\hat{a}\hat{s})}\right)\Omega^2& \\
+I_w^{(f)}\Omega(\dot{\omega}_w^{(fr)} + \dot{\omega}_w^{(fl)})\delta + I_w^{(f)}\Omega(\omega_w^{(fr)} + \omega_w^{(fl)})\dot{\delta} = \tau_y&
\end{aligned} \tag{4.44b}$$

$$(I_{zz} + 2I_{xz}\mu)\dot{\Omega} - \sum_{\hat{a}\hat{s}} I_w^{(\hat{a})}\omega_w^{(\hat{a}\hat{s})}\dot{\phi} - I_{xz}(\ddot{\phi} - 2\Omega\dot{\mu}) - \phi\sum_{\hat{a}\hat{s}} I_w^{(\hat{a}\hat{s})}\dot{\omega}_w^{(\hat{a}\hat{s})} = \tau_z \tag{4.44c}$$

where I_{xx}, I_{yy}, I_{zz} are the principal moments of inertia of the car chassis, I_{xz} is the cross moment of inertia, I_w is the wheel spin inertia moment, $t_v^{(r)} = t_v^{(rl)} = -t_v^{(rr)}$ is the rear half track, $t_v^{(f)} = t_v^{(fl)} = -t_v^{(fl)}$ is the front half track, $b^{(r)} = b$ is the x-axis distance of the car CoM from the rear axle, and $b^{(f)} = -a$ is the x-axis distance of the car CoM from the front axle. Then, the x, y and z components of the net external torque acting on the chassis τ_x, τ_y, and τ_z are:

$$\tau_x = \sum_{\hat{a}\hat{s}} N^{(\hat{a}\hat{s})}(t_v^{(\hat{a}\hat{s})} + h\phi - y_w^{(\hat{a}\hat{s})} + r_t^{(\hat{a})}\gamma_w^{(\hat{a}\hat{s})}) \tag{4.45a}$$

$$\tau_y = F_l^{(r)} b^{(r)} - F_l^{(f)} b^{(f)} + \sum_{\hat{a}\hat{s}} N^{(\hat{a}\hat{s})}\left(-b^{(\hat{a})} + (h - r_t^{(\hat{a})})\mu + x_w^{(\hat{a}\hat{s})}\right) \tag{4.45b}$$

$$\begin{aligned}\tau_z =& \sum_{\hat{a}\hat{s}} F_{tx}^{(\hat{a}\hat{s})}(b^{(\hat{a})}\delta_w^{(\hat{a}\hat{s})} + t_v^{(\hat{a}\hat{s})} + h\phi - y_w^{(\hat{a}\hat{s})} + r_t^{(\hat{a})}\gamma_w^{(\hat{a}\hat{s})})+ \\ & - \sum_{\hat{a}\hat{s}} F_{ty}^{(\hat{a}\hat{s})}\left(t_v^{(\hat{a}\hat{s})}\delta_w^{(\hat{a}\hat{s})} + b^{(\hat{a})} - (h - r_t^{(\hat{a})})\mu - x_w^{(\hat{a}\hat{s})}\right)\end{aligned} \tag{4.45c}$$

where $r_t^{(r)}$ is the rear tyre radius, and $r_t^{(f)}$ is the front tyre radius.

As previously introduced, the suspension kinematics and the nonlinear aerodynamic forces make it no more possible to express the suspension-related degrees of freedom $z_w^{(\hat{a}\hat{s})}$ as explicit function of the other state variables (as it has been done for the massless wheel model in section 4.2.2). Indeed, the suspension force balance equations resulting after neglecting the wheel masses are nonlinear and quite complex. Such balance equations could be included as path constraints in the OCP, however the optimal control software used here to solve the minimum lap time problem (*Pins*, see chapter 3), at the time of this work did not allow to use algebraic constraints. Thus the wheel vertical travels $z_w^{(\hat{a}\hat{s})}$ must be included in the state space model, and the related equations of motion must be calculated. Such equations are here obtained using the Lagrangian approach which, in contrast to the Newton one, allows the suspension links reaction forces to be disregarded. With the generalized force approach, the suspension equations can be derived from:

$$\frac{d}{dt}\left(\frac{\partial K^{(\hat{a}\hat{s})}}{\partial \dot{z}_w^{(\hat{a}\hat{s})}}\right) - \frac{\partial U^{(\hat{a}\hat{s})}}{\partial z_w^{(\hat{a}\hat{s})}} = Q^{(\hat{a}\hat{s})} \tag{4.46}$$

where $\dot{z}_w^{(\hat{a}\hat{s})}$ is the time derivative of the wheel vertical displacement $z_w^{(\hat{a}\hat{s})}$, $K^{(\hat{a}\hat{s})}$ is the kinetic energy of the wheel, $U^{(\hat{a}\hat{s})}$ is the gravitational potential energy, and $Q^{(\hat{a}\hat{s})}$ is the generalized force acting on the wheel. Suspension equations take into account for the suspension kinematics, i.e. that tyre contact point moves fore and aft as well as left and right in the road plane as the suspension moves up and down, so as that tyre lateral and longitudinal forces influence the equilibrium of the suspension. Moreover, each wheel not only translates as the suspension moves, but it also rotates about all the three axes; such rotations influence the in-plane forces generated by the tyre as long as they modify both the tyre sideslip and camber angles. Then the explicit equation form is:

$$\begin{aligned}& m_w^{(\hat{a})}(\ddot{z} - \ddot{z}_w^{(\hat{a}\hat{s})} + b^{(\hat{a})}\ddot{\mu} - t_v^{(\hat{a}\hat{s})}\ddot{\phi} - g) = \\ & - F_s^{(\hat{a}\hat{s})} - N^{(\hat{a}\hat{s})} + F_{tx}^{(\hat{a}\hat{s})}\left(\frac{\partial x_w^{(\hat{a}\hat{s})}}{\partial z_w^{(\hat{a}\hat{s})}} + \frac{\partial \mu_w^{(\hat{a}\hat{s})}}{\partial z_w^{(\hat{a}\hat{s})}} r_t^{(\hat{a})} + \frac{\partial y_w^{(\hat{a}\hat{s})}}{\partial z_w^{(\hat{a}\hat{s})}}\psi_w^{(\hat{a}\hat{s})} + \frac{\partial \psi_w^{(\hat{a}\hat{s})}}{\partial z_w^{(\hat{a}\hat{s})}}\gamma_w^{(\hat{a}\hat{s})} r_t^{(\hat{a})}\right) \\ & - F_{ty}^{(\hat{a}\hat{s})}\left(\frac{\partial y_w^{(\hat{a}\hat{s})}}{\partial z_w^{(\hat{a}\hat{s})}} + \frac{\partial \gamma_w^{(\hat{a}\hat{s})}}{\partial z_w^{(\hat{a}\hat{s})}} r_t^{(\hat{a})} + \frac{\partial x_w^{(\hat{a}\hat{s})}}{\partial z_w^{(\hat{a}\hat{s})}}\psi_w^{(\hat{a}\hat{s})}\right)\end{aligned} \tag{4.47}$$

where $F_s^{(\hat{a}\hat{s})}$ is the suspension force acting on the chassis and counter-reacting on the wheel. The terms $\partial x_w^{(\hat{a}\hat{s})}/\partial z_w^{(\hat{a}\hat{s})}$, $\partial y_w^{(\hat{a}\hat{s})}/\partial z_w^{(\hat{a}\hat{s})}$, $\partial \mu_w^{(\hat{a}\hat{s})}/\partial z_w^{(\hat{a}\hat{s})}$, $\partial \gamma_w^{(\hat{a}\hat{s})}/\partial z_w^{(\hat{a}\hat{s})}$ and $\partial \psi_w^{(\hat{a}\hat{s})}/\partial z_w^{(\hat{a}\hat{s})}$ are those related to suspension kinematics and, as it can be noticed, they determine how the tyre longitudinal and lateral forces contribute to suspension motion. When equation (4.47) is derived from (4.46), the inertial terms related to the time derivative of the secondary wheel plate variables in $\mathbf{u}_{as} = \left[x_w^{(\hat{a}\hat{s})}; y_w^{(\hat{a}\hat{s})}; \gamma_w^{(\hat{a}\hat{s})}; \psi_w^{(\hat{a}\hat{s})}; \mu_w^{(\hat{a}\hat{s})}\right]$ have been neglected as long as they do not have a relevant physical effect on suspension dynamics. Indeed such terms are related to small inertial forces (less than 0.1 times the inertial force related to $\ddot{z_w}^{(\hat{a}\hat{s})}$) that do not play an important role in time simulations, as demonstrated in section 4.2 with the massless-wheel model. On the contrary, the terms related to the ratio between variables in $\mathbf{u}_{as}$ and $z_w^{(\hat{a}\hat{s})}$ are required to take into account the anti-lift, anti-squat or scrub behaviour of the suspension and are relevant even in determining the stationary trim of the suspension or of the vehicle [6, 64].

Wheel spin motion is governed by tyre forces and the driver's braking or driving input torques. Euler equations for the wheel spins are:

$$I_w^{(\hat{a})} \dot{\omega}_w^{(\hat{a}\hat{s})} = \tau_{wd}^{(\hat{a}\hat{s})} - F_{tx}^{(\hat{a}\hat{s})}(r_t^{(\hat{a})} - z_w^{(\hat{a}\hat{s})} - z - b^{(\hat{a})}\mu + t_v^{(\hat{a}\hat{s})}\phi) \tag{4.48}$$

where $I_w^{(\hat{a})}$ is wheel inertia moment around the spin axis (including also half of the axle and powertrain inertias in the rear wheels), $r_t^{(\hat{a})}$ is the tyre radius, $F_{tx}^{(\hat{a}\hat{s})}$ is the tyre longitudinal force and $\tau_{wd}^{(\hat{a}\hat{s})}$ is the torque delivered to the wheel. In the above equation the inertial terms related to the fact that the wheel spin axis is not fixed but moves with and with respect the chassis have been neglected because their expression is complex but their effect is negligible.

4.3.1.3 Forces

The multibody car model here presented is fully described by the chassis Newton (4.43) and Euler (4.44) equations, together with the suspension (4.47) and wheel spin (4.48) ones, however the forces that appears in these equations still have to be made explicit. In this section the suspension, tyre and aerodynamic forces, together with the wheel driving torques are expressed in terms of the model state variables and controls.

The four torque inputs $\tau_{wd}^{(\hat{a}\hat{s})}$ depend on one variable, the overall driving torque τ_d, exactly as it has been done for the 14 dof GT car model. Thus wheel torques $\tau_{wd}^{(\hat{a}\hat{s})}$ are given by equations (4.11) together with (4.12). It can be noticed that the engine torque is only positive, in other words the engine brake torque is neglected; this is a consequence of the lack of experimental data regarding the negative torque exerted by the engine at zero throttle.

Suspensions forces $F_s^{(\hat{a}\hat{s})}$ acting on the chassis and counter-reacting on each wheel are the sum of elastic (spring), damper, anti-roll bar $F_{sa}^{(\hat{a}\hat{s})}$ and bump rubber $F_{sp}^{(\hat{a}\hat{s})}$ forces:

$$F_s^{(\hat{a}\hat{s})} = K_{ss}^{(\hat{a}\hat{s})} z_w^{(\hat{a}\hat{s})} + K_{sd}^{(\hat{a}\hat{s})} \dot{z}_w^{(\hat{a}\hat{s})} + F_{sa}^{(\hat{a}\hat{s})} + F_{sp}^{(\hat{a}\hat{s})} \tag{4.49}$$

where $K_{ss}^{(\hat{a}\hat{s})}$ is the elastic force stiffness exerted by the torsion bars (front suspensions) or coil springs (rear suspensions), $K_{sd}^{(\hat{a}\hat{s})}$ is the damping coefficient and $z_w^{(\hat{a}\hat{s})}$ is the wheel travel. The elastic and damping forces are expressed by a linear relationship as long as the velocity ratio between the wheel travel and torsion bar rotation angle or damper travel is constant in the working range of the suspensions. The anti-roll bar forces $F_{sa}^{(\hat{a}\hat{s})}$ depend on the difference between right and left wheel displacement:

$$\begin{aligned} F_{sa}^{(rr)} &= K_a^{(r)}(z_w^{(rr)} - z_w^{(rl)}) = -F_{sa}^{(rl)} \\ F_{sa}^{(fr)} &= K_a^{(f)}(z_w^{(fr)} - z_w^{(fl)}) = -F_{sa}^{(fl)} \end{aligned} \tag{4.50}$$

where $K_a^{(r)}$ and $K_a^{(f)}$ are the rear and front reduced anti-roll bar stiffness. The last force $F_{sp}^{(\hat{a}\hat{s})}$ in equation (4.49) is the force due to the bump rubbers, which prevent the suspensions from excessive travel. The bump rubber forces are given by splines used to fit the experimental force versus deflection curve data. Both the experimental data and the fitting splines are shown in figure 4.9, together with the spring forces. Figure 4.9 highlights that the rear bump rubber force equals the rear spring force at a wheel travel of only $6mm$, and the front bump rubber force equal the front torsion bar one at a wheel travel of $9mm$. For higher wheel displacement, the bump rubber forces significantly exceed the spring/torsion bar forces and this makes the suspension rates highly non-linear.

In a GP2/F1 car model, tyres cannot be considered completely rigid as long as their radial stiffness is comparable to that of suspensions. The tyre manufacturer provides a specific formula to calculate the stationary tyre radial deformation as function of the tyre load, spin, pressure, camber and lateral force:

$$\xi^{(\hat{a}\hat{s})} = \frac{N^{(\hat{a}\hat{s})}}{p_1 \rho_p^{(\hat{a}\hat{s})} + a_1 \omega_w^{2\,(\hat{a}\hat{s})} + a_2 \omega_w^{(\hat{a}\hat{s})} + a_3 + c_1 \left| \gamma_w^{(\hat{a}\hat{s})} \right| + f_1 {F_{ty}^{(\hat{a}\hat{s})}}^2 / N^{(\hat{a}\hat{s})}} + b_1 \omega_w^{2\,(\hat{a}\hat{s})} + b_2 \omega_w^{(\hat{a}\hat{s})} \tag{4.51}$$

where $N^{(\hat{a}\hat{s})}$ is the tyre load, $\xi^{(\hat{a}\hat{s})}$ the tyre radial deformation, $\omega_w^{(\hat{a}\hat{s})}$ the wheel spin, $\rho_p^{(\hat{a}\hat{s})}$ is the tyre pressure, $p_1, a_1, a_2, a_3, c_1, f_1, b_1$ and b_2 are constant coefficients. As long as tyre radial deformations are determined by the chassis and suspensions trim, while tyre loads are not, the previous formula has been numerically inverted in order to obtain the tyre loads

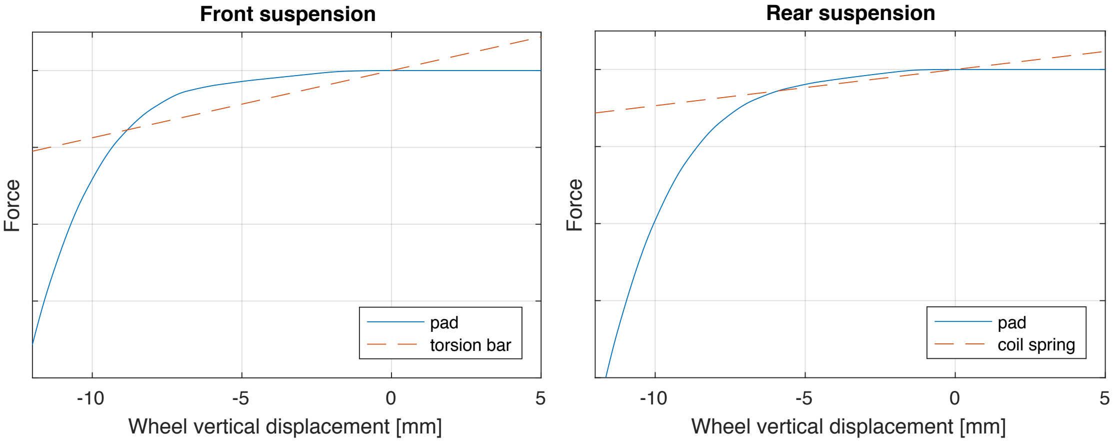


Figure 4.9: Bump rubber force for the rear and front suspensions. The orange dashed line show the torsion bar (front) or coil spring (rear) forces. The net force exerted on the wheel is significantly non-linear for a wheel travel of $6mm$ (front) and $4mm$ (rear).

as function of the other variables:

$$N^{(\hat{a}\hat{s})} = f^{+}(n_d^{(\hat{a})}\xi^{(\hat{a}\hat{s})} + n_{do}^{(\hat{a})}\xi^{(\hat{a}\hat{s})}\omega_w^{(\hat{a}\hat{s})} + n_o^{(\hat{a})}\omega_w^{2\,(\hat{a}\hat{s})} + n_y^{(\hat{a})}F_{ty}^{(\hat{a}\hat{s})^2} + n_c^{(\hat{a})}\dot{\xi}) \tag{4.52}$$

where $n_d^{(\hat{a})}$, $n_{do}^{(\hat{a})}$, $n_o^{(\hat{a})}$ and $n_y^{(\hat{a})}$ are the coefficients used to fit the data calculated using (4.51). The camber dependence has been neglected, and the pressure has been considered to be constant at $19psi$ for the rear tyre, and $20psi$ for the front one. A damping coefficient n_c has been added to simulate the tyre radial damping. The f^{+} function ensures that the tyre load never becomes negative. The tyre radial deformations are determined by the chassis and suspension trims, and they are given by the same formulas used for the 14 dof GT-car model (4.10). With the approach that has been adopted, lateral and longitudinal load transfers are automatically calculated according to the model state variables as well as suspension characteristics. Longitudinal $F_{tx}^{(\hat{a}\hat{s})}$ and lateral $F_{ty}^{(\hat{a}\hat{s})}$ tyre forces are given by a Magic Formula Tyre model [70], as functions of tyre load, longitudinal slip, sideslip and camber angles. Moreover a relaxation equation for every tyre force has been added because it is known that tyre forces raise with a certain time delay with respect to input variables, especially in the case of lateral forces [70]. The equations governing the tyre force relaxations are similar to the one used for the 14 dof GT-car model (4.16), (4.17), except for the different speed of the tyre contact point:

$$\begin{aligned}\frac{\sigma_{tx}^{(\hat{a})}}{V_{tx}^{(\hat{a}\hat{s})}}\dot{F}_{tx}^{(\hat{a}\hat{s})} + F_{tx}^{(\hat{a}\hat{s})} &= F_{tx,magic}^{(\hat{a})}(N^{(\hat{a}\hat{s})}, \kappa_t^{(\hat{a}\hat{s})}, \lambda_t^{(\hat{a}\hat{s})}, \phi_w^{(\hat{a}\hat{s})}) \\ \frac{\sigma_{ty}^{(\hat{a})}}{V_{tx}^{(\hat{a}\hat{s})}}\dot{F}_{ty}^{(\hat{a}\hat{s})} + F_{ty}^{(\hat{a}\hat{s})} &= F_{ty,magic}^{(\hat{a})}(N^{(\hat{a}\hat{s})}, \kappa_t^{(\hat{a}\hat{s})}, \lambda_t^{(\hat{a}\hat{s})}, \phi_w^{(\hat{a}\hat{s})})\end{aligned} \tag{4.53}$$

where $\phi_w^{(\hat{a}\hat{s})}$ is the wheel camber angle, σ_{tx} and σ_{ty} are the relaxation lengths of the tyre in the longitudinal and lateral direction, $F_{tx,magic}$ and $F_{ty,magic}$ are the stationary Magic Formula tyre forces, $\kappa_t^{(\hat{a}\hat{s})}$, $\lambda_t^{(\hat{a}\hat{s})}$, $\phi_w^{(\hat{a}\hat{s})}$ are respectively the tyre longitudinal slip, sideslip angle and camber angle and $V_{tx}^{(\hat{a}\hat{s})}$ is the tyre contact point longitudinal speed. The slip quantities necessary to calculate the tyre forces can be obtained as function of the model state variables as shown below:

$$\begin{aligned}V_{tx}^{(\hat{a}\hat{s})} =& V + \Omega t_v^{(\hat{a})} + \dot{\mu}(h - r_t^{(\hat{a})} + \dot{x}_w^{(\hat{a}\hat{s})} + \Omega\phi h \\ & + \Omega\gamma_w^{(\hat{a}\hat{s})} r_t^{(\hat{a})} - \Omega b^{(\hat{a})}(\psi_w^{(\hat{a}\hat{s})} + \delta_{w0}^{(\hat{a}\hat{s})}) - \Omega y_w^{(\hat{a}\hat{s})} \\ V_{ty}^{(\hat{a}\hat{s})} =& \Omega(x_w^{(\hat{a}\hat{s})} - b) - \dot{\phi}h - \dot{\gamma}_w^{(\hat{a}\hat{s})} r_t^{(\hat{a})} + \dot{y}_w^{(\hat{a}\hat{s})} - V\lambda \\ & + (h - r_t^{(\hat{a})})\Omega\mu - (V - \Omega t_v^{(\hat{a})})(\psi_w^{(\hat{a}\hat{s})} + \delta_{w0}^{(\hat{a}\hat{s})}) \\ V_{tr}^{(\hat{a}\hat{s})} =& \dot{\mu}_w^{(\hat{a}\hat{s})} r_{tr}^{(\hat{a})} - \omega_w^{(\hat{a}\hat{s})}(r_t^{(\hat{a})} + z + z_w^{(\hat{a}\hat{s})} + b^{(\hat{a})}\mu - t_v^{(\hat{a})}\phi) \\ \kappa^{(\hat{a}\hat{s})} =& \frac{V_{tr}^{(\hat{a}\hat{s})}}{V_{tx}^{(\hat{a}\hat{s})}} - 1 \\ \lambda^{(\hat{a}\hat{s})} =& -\frac{V_{ty}^{(\hat{a}\hat{s})}}{V_{tx}^{(\hat{a}\hat{s})}} \\ \phi_w^{(\hat{a}\hat{s})} =& \phi + \phi_{w0}^{(\hat{a}\hat{s})} + \gamma_w^{(\hat{a}\hat{s})} \\ \delta_w^{(\hat{a}\hat{s})} =& \delta_{w0}^{(\hat{a}\hat{s})} + \psi_w^{(\hat{a}\hat{s})}\end{aligned} \tag{4.54}$$

where ϕ_w and δ_w are respectively the wheel camber and steer angle (measured in the road plane), ϕ_{w0} and δ_{w0} are the wheel camber and steer angle at the car nominal trim, V_{ty} and V_{tr} are the tyre contact point lateral and rolling speeds, r_{tr} is the tyre rolling radius. Moreover in the computation of $\lambda^{(rr)}$ the approximation $\tan(x) \approx x$ holds.

The rolling resistance of each tyre is proportional to the tyre load. Thus, the total rolling resistance force has been approximated with the following expression:

$$R_t = K_{tR}^{(r)}\left(Mg\frac{a}{w} + F_l^{(r)}\right) + K_{tR}^{(f)}\left(Mg\frac{b}{w} + F_l^{(f)}\right) \tag{4.55}$$

where $K_{tR}^{(r)}$ and $K_{tR}^{(f)}$ are the rear and front tyre rolling resistance coefficients.

Aerodynamic forces that act on the car are as follows:

$$\begin{aligned} F_d &= \frac{1}{2}\rho C_d(h^{(r)}, h^{(f)})V^2 \\ F_l^{(r)} &= -\frac{1}{2}\rho C_l^{(r)}(h^{(r)}, h^{(f)})V^2 \\ F_l^{(f)} &= -\frac{1}{2}\rho C_l^{(f)}(h^{(r)}, h^{(f)})V^2 \end{aligned} \tag{4.56}$$

where F_d is the drag force, $F_l^{(r)}$ and $F_l^{(f)}$ are respectively the rear and front aerodynamic downforce, ρ is air density, C_d, $C_l^{(r)}$, $C_l^{(f)}$ are the drag, rear lift and front lift coefficients that depend on the rear $h^{(r)}$ and front $h^{(f)}$ ride heights as third-degree polynomials given by the car manufacturer. The coefficients used in such polynomials are those provided by the car manufacturer and depend on the wing regulations. The average front and rear ride heights can be easily calculated as function of z and μ on the basis of elementary geometrical considerations:

$$h^{(r)} = h_0^{(r)} + \mu b - z \qquad\qquad h^{(f)} \quad = h_0^{(f)} - \mu a - z \tag{4.57}$$

where $h_0^{(r)}$ and $h_0^{(f)}$ are respectively the rear and front ride height nominal values.

4.3.1.4 Steady State Analysis

The model described so far can be utilized to analyse the vehicle steady state trim and to verify the influence of suspension design on load transfers. The steady state trim of the car can be calculated solving an equation system including equations (4.43) (4.44) (4.47) (4.48), with the conditions that all the time derivatives of the variables are zero, i.e. $\dot{V} = \dot{\lambda} = \dot{\Omega} = \dot{\phi} = \dot{\mu} = \dot{z} = \dot{z}_w^{(\hat{a}\hat{s})} = \dot{\omega_w}^{(\hat{a}\hat{s})} = 0$, thus solving an algebraic system. In order for such equations system to be determined, two auxiliary conditions have to be imposed (due to the presence in the equations of two unknown controls):

$$\begin{cases} V = V_0 \\ \Omega = \Omega_0 \end{cases} \tag{4.58}$$

where V_0 and Ω_0 can be arbitrarily chosen. The solution gives the car stationary trim for different values of the speed and yaw rate. The longitudinal acceleration can be artificially introduced in the steady state equations by adding a horizontal virtual gravity a_x, acting on all the bodies. The numeric data used to feed the model is reported in table 4.3.1.4.

Table 4.6: GP2 data used for simulations.

symbol	value	units	description
g	9.81	m/s^2	gravitational acceleration
ρ	1.2	kg/m^3	air density
h	0.31	m	centre of gravity (CoG) height
$b = b^{(r)}$	1.34	m	x-axis distance between the rear axle and the vehicle CoG
$a = -b^{(f)}$	1.78	m	x-axis distance between the front axle and the vehicle CoG
w	3.12	m	wheelbase
$t_v^{(f)} = t_{fl} = -t_{fr}$	0.739	m	front half track
$t_v^{(r)} = t_{rl} = -t_{rr}$	0.708	m	rear half track
$h_0^{(r)}$	6×10^{-2}	m	nominal rear ride height
$h_0^{(f)}$	2×10^{-2}	m	nominal front ride height
γ	0.62	-	front braking bias
ψ_{rr0}	-0.04	deg	rear right wheel nominal yaw angle
γ_{rr0}	-0.5	deg	rear right wheel nominal camber angle
ψ_{fr0}	0.09	deg	front right wheel nominal yaw angle
γ_{fr0}	-2.7	deg	front right wheel nominal camber angle
m	700	kg	vehicle mass (rider included)
I_{xx}	200	kgm^2	roll moment of inertia
I_{yy}	1000	kgm^2	pitch moment of inertia
I_{zz}	1100	kgm^2	yaw moment of inertia
I_{xz}	0	kgm^2	mixed moment of inertia
$m_w^{(r)}$	30.4	kg	rear wheel mass
$m_w^{(f)}$	24.4	kg	front wheel mass
$I_w^{(r)}$	1.55	kgm^2	rear wheel spin inertia (including half of rear axle)
$I_w^{(f)}$	1	kgm^2	front wheel spin inertia
$n_d^{(r)}$	2.03×10^{-5}	N/m	rear tyre load coefficient
$n_d^{(f)}$	1.96×10^{-5}	N/m	front tyre load coefficient

symbol	value	units	description
$n_{do}^{(r)}$	728	Ns/m	rear tyre load coefficient
$n_{do}^{(f)}$	473	Ns/m	front tyre load coefficient
$n_{o}^{(r)}$	5.85×10^{-2}	Ns^2	rear tyre load coefficient
$n_{o}^{(f)}$	1.57×10^{-2}	Ns^2	front tyre load coefficient
$n_{y}^{(r)}$	-1.39×10^{-5}	N^{-1}	rear tyre load coefficient
$n_{y}^{(f)}$	-1.9×10^{-5}	N^{-1}	front tyre load coefficient
$n_{c}^{(r)}$	500	Ns/m	rear tyre load damping stiffness
$n_{c}^{(f)}$	500	Ns/m	front tyre load damping stiffness
$K_{tR}^{(r)}$	0.01	-	rear tyre rolling resistance coefficient
$K_{tR}^{(f)}$	0.01	-	front tyre rolling resistance coefficient
$r_{tr}^{(r)}$	0.31	m	rear tyres radius
$r_{tr}^{(f)}$	0.31	m	front tyres radius
$\sigma_{tx}^{(r)} = \sigma_{tx}^{(f)}$	0.1	m	tyres lateral relaxation length
$\sigma_{ty}^{(r)} = \sigma_{ty}^{(f)}$	0.1	m	tyres longitudinal relaxation length

Figure 4.10 shows the tyre loads versus the centripetal acceleration $a_c = \Omega V$ at a speed of $V = 40m/s$, both for a fixed longitudinal acceleration $a_x = g$ (plots in the top row) and $a_x = -g$ (plots in the bottom row). In order to highlight the influence of suspension design on vehicle trim, such figure compares the tyre loads of the GP2 car (normal model) with those calculated for a car that is exactly the same of the one studied so far, but where the wheel plane moves only vertically, i.e. $x_w^{(\hat{a}\hat{s})} = y_w^{(\hat{a}\hat{s})} = \gamma_w^{(\hat{a}\hat{s})} = \mu_w^{(\hat{a}\hat{s})} = 0$ (simple model).
The results show that for a longitudinal acceleration $a_x = g$ the tyre loads are almost equal for the two models, even though there are differences of $\approx 2mm$ in the suspension trim (which are not shown) since the simple model does not include any anti-squat behavior. This is not a general result because it depends on the suspension design, car trim and tyre forces, and a different scenario can lead to different outcomes. Indeed, when the car is in stationary braking conditions (i.e. braking at constant negative acceleration), some differences in the results between the two models arise, as shown in the two plots at the bottom of figure 4.10: in the full model the rear axle lateral load transfer is higher, while the front one is lower (this allows greater negative accelerations before locking the front wheels). More precisely, for a centripetal acceleration $a_c = 15.4m/s^2$, the lateral load transfer at the rear axle for the full and simple models are respectively of $2016N$ and $1853N$,

with a relative difference of approximately 8%. This steady state analysis confirms that the suspension kinematics influences the tyre loads even in stationary conditions. Moreover, since load transfers are a key feature that influences the manoeuvrability of four-wheeled vehicles, suspension kinematics should not be disregarded in minimum lap time simulations as long as really accurate results are needed.

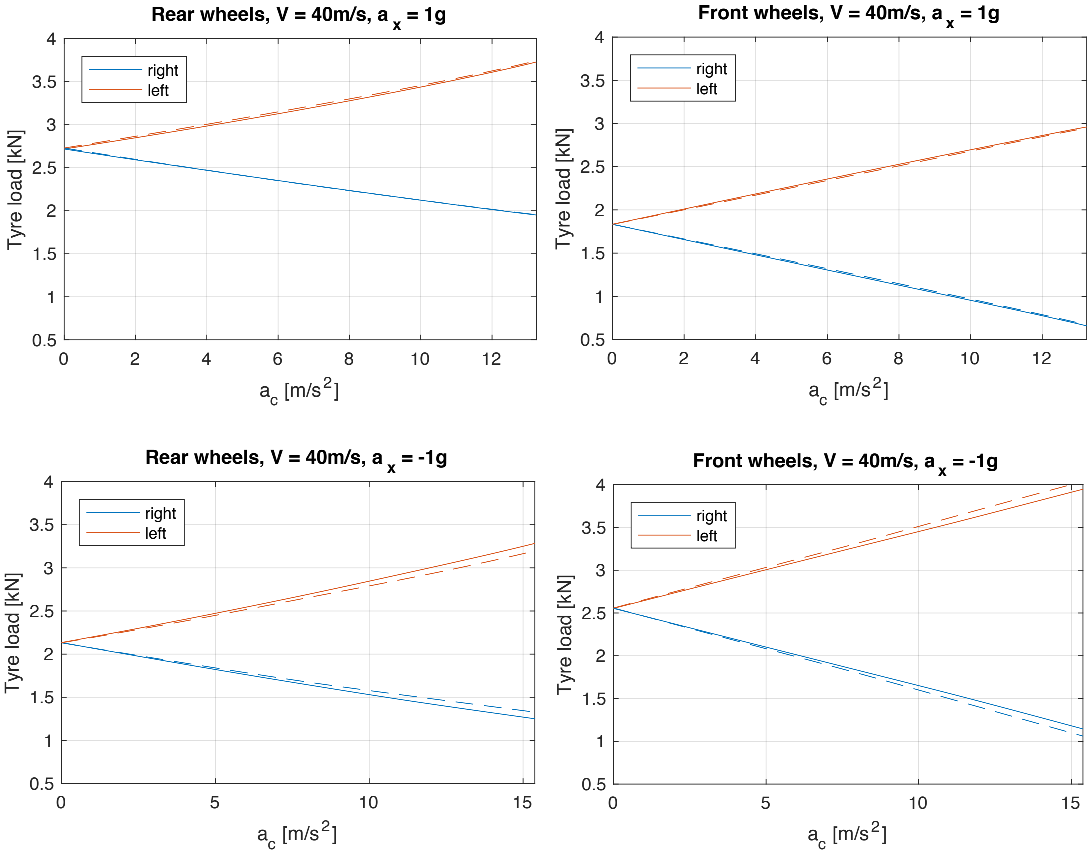


Figure 4.10: Static tyre loads at a speed of $40m/s$ and longitudinal acceleration $a_x = 1g$ (top) and $a_x = -1g$ (bottom). Continuous lines refer to the normal model, dashed ones to the simple one.

4.3.2 State space and OCP formulation

Some of the state variables used in the model equations (4.43) to (4.48) appear with their second order derivatives. The equations may be immediately reduced to an ODE system by introducing auxiliary variables for the relevant chassis and suspension speeds:

$$\dot{z} = z_{dot} \quad \dot{\phi} = \phi_{dot} \quad \dot{\mu} = \mu_{dot} \quad \dot{z}_w^{(\hat{a}\hat{s})} = z_{w,dot}^{(\hat{a}\hat{s})} \tag{4.59}$$

At this point, equations (4.43) to (4.59), plus the road tracking equations (4.31) completely describe car dynamics as a system of 32 first order differential equations with as many state variables and 2 inputs, respectively the overall driving torque τ_d and the steering rack displacement y_{sr}.

The optimal control formulation is very similar to the one already described in section 4.2.5, thus it is only briefly presented here. The two driver inputs τ_d and y_{sr} have been included to the state variables, while the controls have been moved to their time derivatives, j_τ and j_y, satisfying the following equations:

$$j_y = \dot{y}_{sr} \qquad j_\tau = \dot{\tau}_d \tag{4.60}$$

Moreover, the state variable s is made the independent variable of the OCP by performing the time to space transformation described in section A.

In conclusion, optimal control states include 33 variables:

$$\boldsymbol{x} = \Big[n; \alpha; V; \lambda; z; z_{dot}; \phi; \phi_{dot}; \mu; \mu_{dot}; \Omega; z_w^{(\hat{a}\hat{s})}; z_{w,dot}^{(\hat{a}\hat{s})}; \omega_w^{(\hat{a}\hat{s})}; F_{ty}^{(\hat{a}\hat{s})}; F_{tx}^{(\hat{a}\hat{s})}; y_{sr}; \tau_d\Big] \tag{4.61}$$

plus two inputs:

$$\boldsymbol{u} = \big[j_y; j_\tau\big] \tag{4.62}$$

and as many implicit first order differential equations.

The OC target to be minimised is equal to the lap time (the integral of $1/\dot{s}$, see equation (4.31a)) and the OCP boundary conditions are cyclic. OC constraints include the road boundaries limit, the engine power limit, the front tyre negative slip limit and the control bounds, exactly as it has been done in section 4.2.5.

The *Pins* software described in chapter 3.2.1 has been chosen to solve the minimum lap time OCP.

4.3.3 Lap time simulation and model validation

Optimal control simulation has been carried out on the circuit of Montmelò in Barcelona. The simulation, that required to solve a system of nearly 1.5×10^5 algebraic equations and as many discrete variables, took approximately 26 minutes to be computed on an *Intel Core i7* based desktop computer, suggesting the proposed model, together with the used software (*Pins*, see section 3.2.1), is very efficient. In the first part of this section the car model will be validated through a comparison between simulation results and the experimental data acquired in a qualifying lap of a driver in 2012 GP2 season, while the second part will be focused on the analysis of the car dynamics.

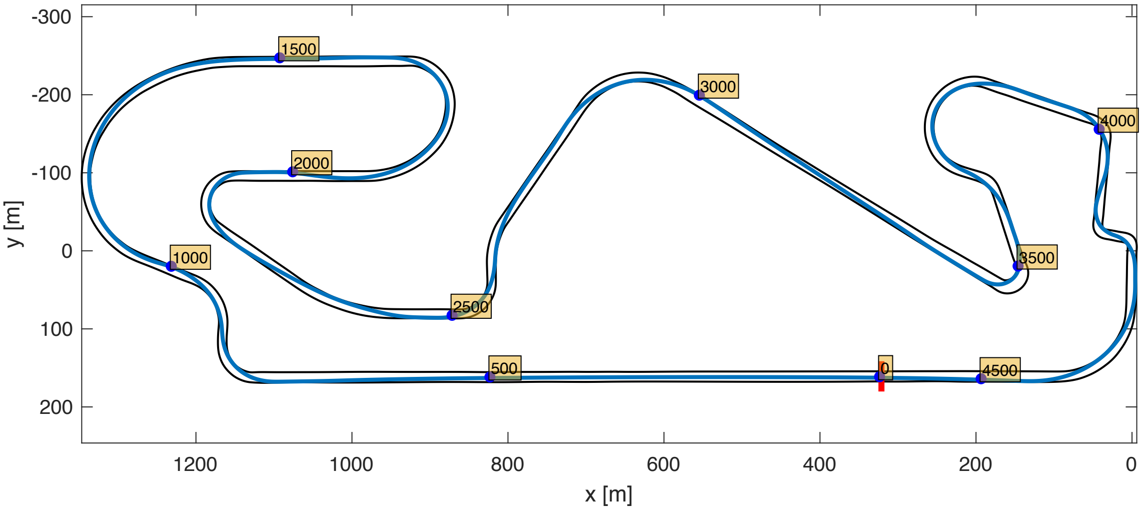


Figure 4.11: Optimal trajectory on Montmelò circuit. Numbers within ochre boxes indicate the value of the road centre line abscissa in meters.

Figure 4.11 shows the car trajectory resulting from the simulation; the road geometry has been reconstructed with the procedure described in appendix B. As expected, the ideal driver tends to smooth the trajectory through corners in order to achieve the minimum time. Figure 4.12 compares the simulated and the experimental speed profiles; there is good agreement between simulation and telemetry especially in the first two thirds of the track, while in the last sector the simulated speed is slightly higher than the real one in correspondence of the corner apex points. Indeed the simulated lap time ($91.287s$) is lower than the real one ($91.600s$) by approximately $0.3s$. It is known that in a qualifying lap at the Montmelò circuit, in the third sector rear tyres are very warm and their performance tends to decrease, thus this phenomenon might generate the discrepancies between simulated and real speed profile that arises in that part of the track. In figure 4.12 both longitudinal a_x and lateral a_y accelerations are also reported, and it can be observed that the simulated ones are close to the real ones; a_x and a_y bounds are of approximately $-40m/s^2 < a_x < 20m/s^2$ and $|a_y| < 30m/s^2$. The resulting g-g diagram, which is represented in the bottom-left corner of figure 4.13, highlights an ellipsoidal shape for positive accelerations, while it has a remarkable triangular shape for negative ones; this difference arises mainly because of the front inner wheel locking that occurs when braking while steering, limiting the lateral acceleration. The maximum accelerations that the car can withstand highly depend on the speed, as long as the aerodynamic downforce increases with the square of the speed. The influence of the speed on the acceleration limits is also shown in figure 4.13: a_x is the one which varies more with the speed, passing from $-20m/s^2 < a_x < 20m/s^2$ at a speed of

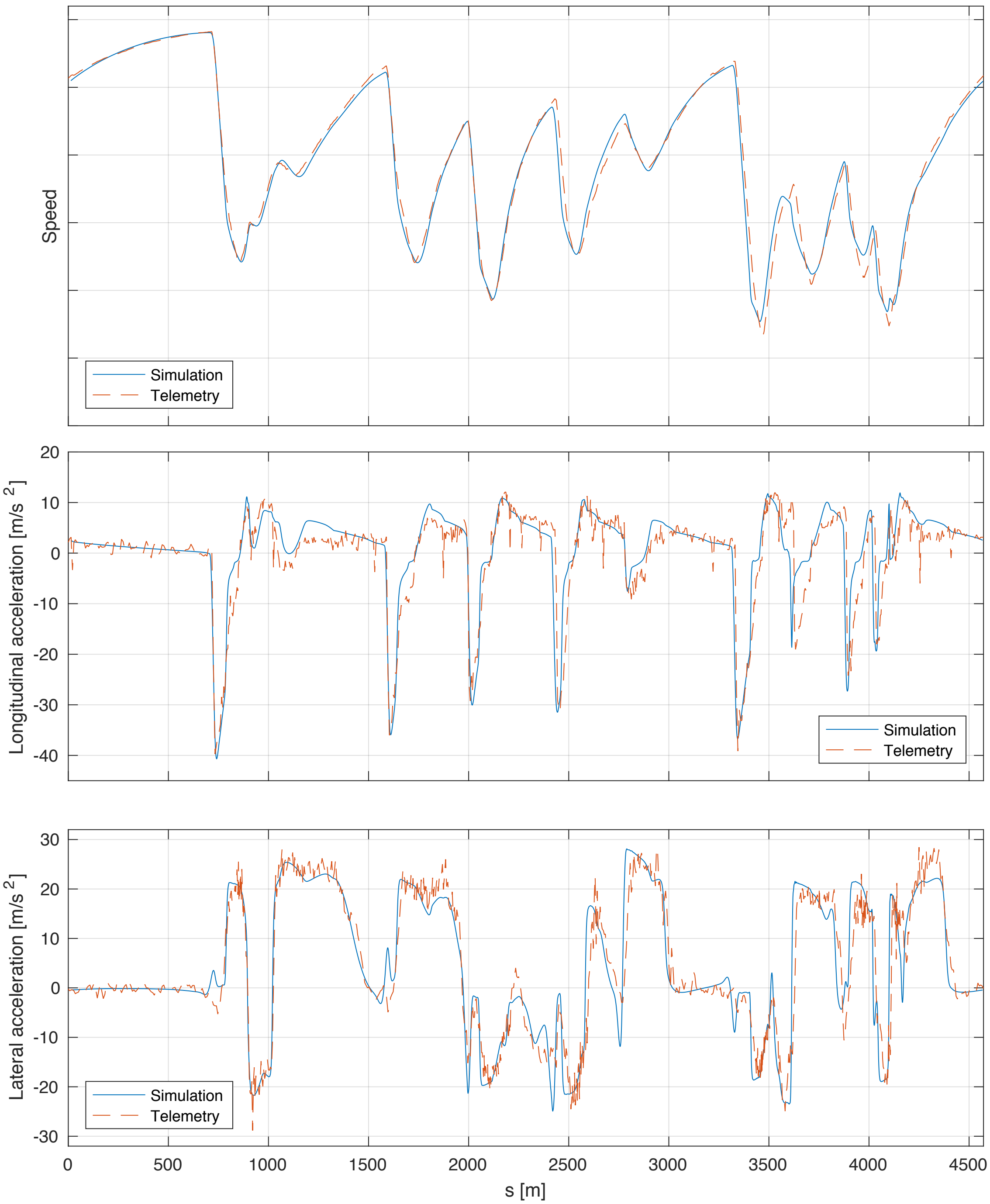


Figure 4.12: Speed (top), longitudinal (centre) and lateral (bottom) accelerations versus distance travelled. Blue continuous lines refer to simulation outcomes, orange dashed ones to telemetry data.

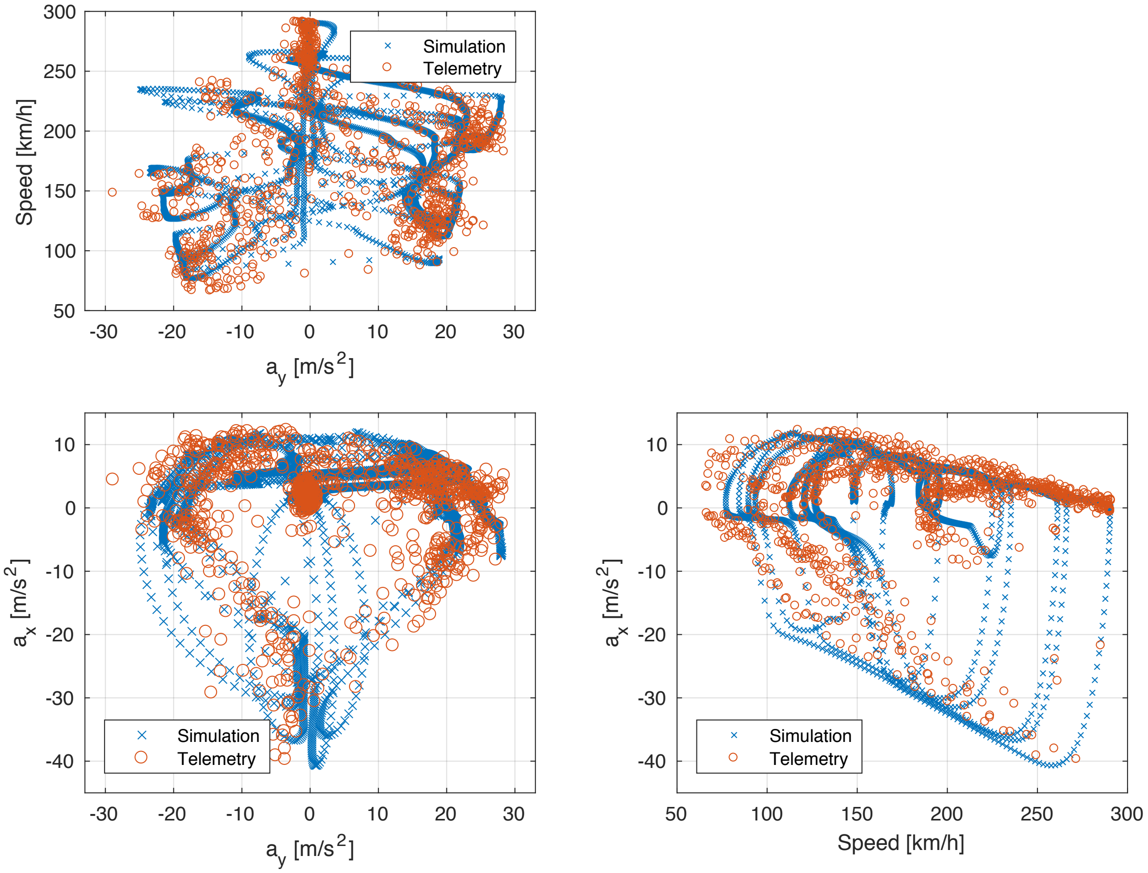


Figure 4.13: Accelerations versus car speed (top left and bottom right) and g-g diagram (bottom left). Blue crosses refer to simulation outcomes, orange circles to telemetry data. The car model reproduces well the real accelerations envelope, moreover the increase of accelerations bounds with speed due to aerodynamic down-force is highly noticeable and well captured.

$\approx 100km/h$ up to $-40m/s^2 < a_x < 5m/s^2$ at a speed of $\approx 250km/h$, while the lateral acceleration increases only from $-20m/s^2 < a_y < 20m/s^2$ to $-28m/s^2 < a_y < 28m/s^2$ for the same speed range. From speed and acceleration comparisons, it can be stated that the model developed in this chapter is able to well reproduce the dynamics of a GP2 car. Some differences are clearly present, but they could be reduced with a better measurement of some parameters affecting the performance, first of all tyre characterisation and road geometry.

The motion of the chassis along the track is shown in figure 4.14, where the vertical displacement z, together with roll ϕ and pitch μ angles are reported. All these quantities are small, indeed z is comprised between $0mm$ and $25mm$, the roll angle is lower than $1°$ and the pitch angle is never larger than $0.5°$. The very limited chassis displacements are consequence of the high suspension stiffness, which is even greater than that of the tyres due to the presence of suspension bump rubbers. Indeed, suspension travels are in the range $-10mm < z_w^{(\hat{a}\hat{s})} < 7mm$ (where negative values correspond to a compressed suspension), while tyre radial deformation spans in $0mm < \xi^{(\hat{a}\hat{s})} < 30mm$, as it can be observed in figure 4.15. Within such limited range of suspension travel and tyre deformation, tyre loads vary from almost 0 to $6000N$.

Even if chassis and suspensions motions are so limited, they noticeably affect both aerodynamic forces and load transfers. Figure 4.14 shows also the aerodynamic drag and downforce together with the ride heights and the downforce balance (i.e. the front by total downforce ratio $F_l^{(f)}/(F_l^{(f)} + F_l^{(r)})$). In such figure, the aerodynamic force trends resemble that of the speed, due to the strict dependence of the former on the square of the latter. However, as long as ride heights change along the track, varying in the range $-5mm < h^{(f)} < 21mm$ (front ride height) and $40 < h^{(r)} < 60$ (rear ride height), the downforce balance also changes. Indeed the downforce balance generally increases with the speed, moreover it bumps up in correspondence of high braking manoeuvres. It can be observed that, when the car withstands high negative accelerations, the front ride height decreases while the rear one increases due to the chassis pitch, thus affecting aero balance. Moreover, the minimum value of the front ride height is negative ($-5mm$) which might seem non-realistic; however, it should be considered that the leading edge of the skid plane is located well behind the front axle (that is where the front ride height is calculated), and that the skid plane is quite flexible, thus slightly negative values for the front ride heights are very likely to be reached when the car is bottoming on the road surface.

As previously said, chassis and suspension motions not only influence aerodynamic forces, but also load transfers. The lateral load transfer at the rear axle ($N_{lat}^{(r)} \equiv N^{(rr)} - N^{(rl)}$) versus that at the front one ($N_{lat}^{(f)} \equiv N^{(fr)} - N^{(fl)}$) resulting from the simula-

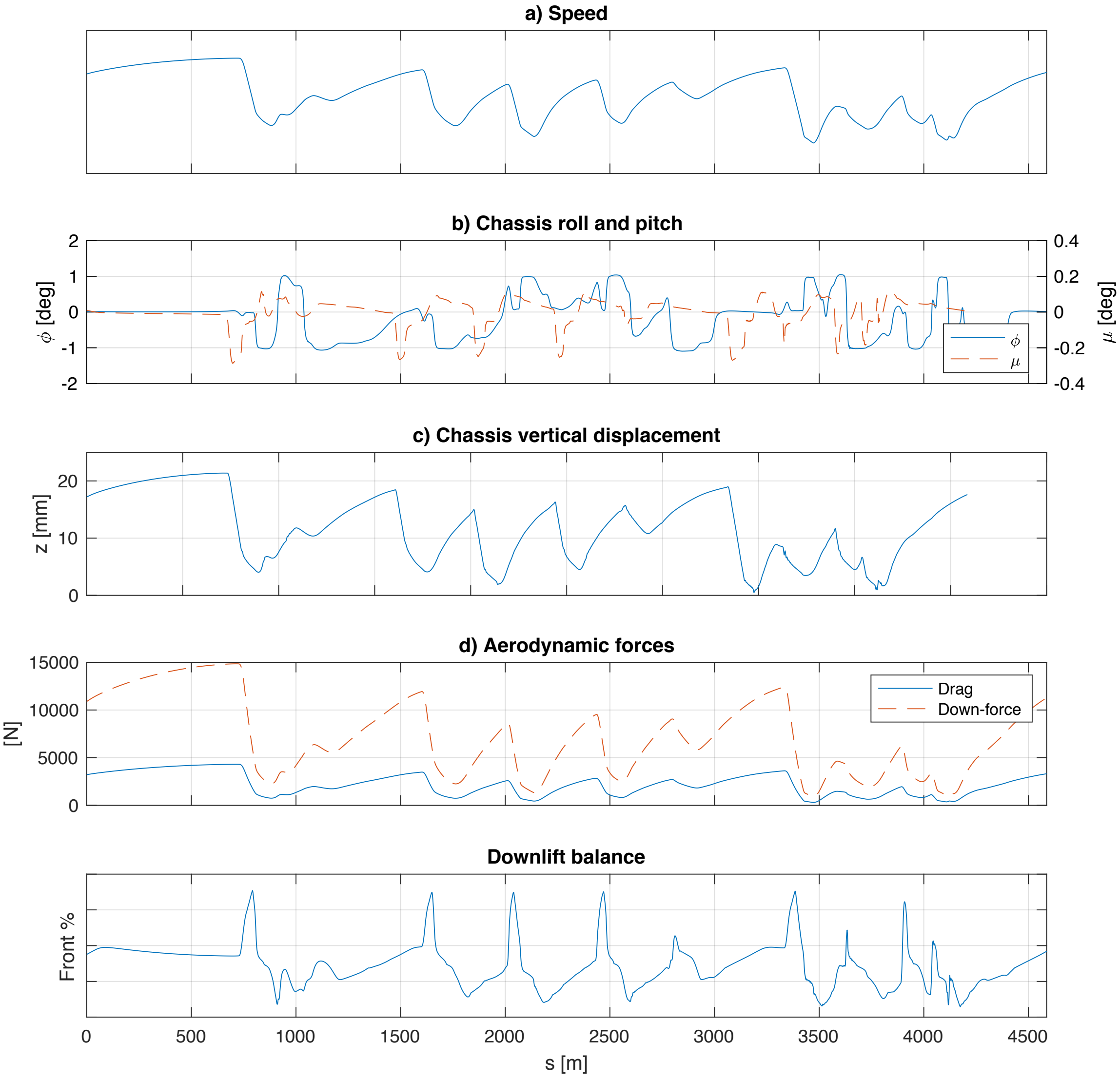


Figure 4.14: From top to bottom: simulated speed (a), chassis roll ϕ and pitch μ angles (b), chassis vertical displacement z (c), aerodynamic down-forces $F_l^{(f)} + F_l^{(r)}$ and drag D (d), front down-force balance $F_l^{(f)}/(F_l^{(f)} + F_l^{(r)})$ (e).

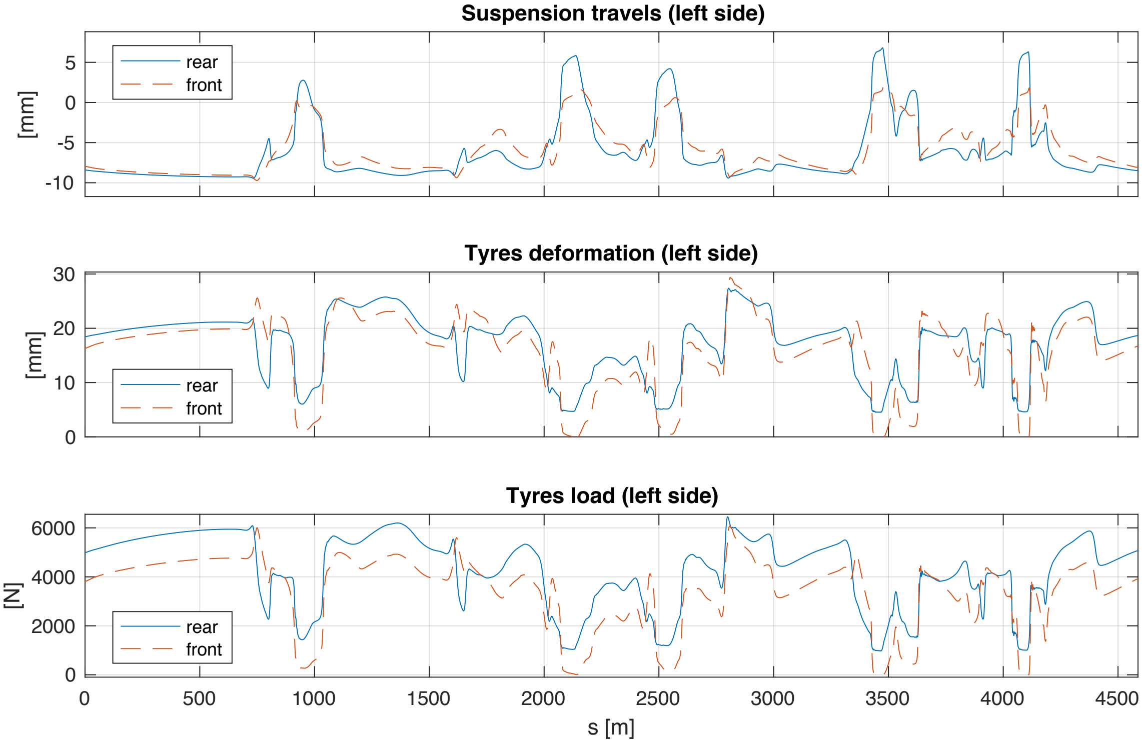


Figure 4.15: Simulated suspension travels, tyre radial deformations and tyre loads (left side of the car).

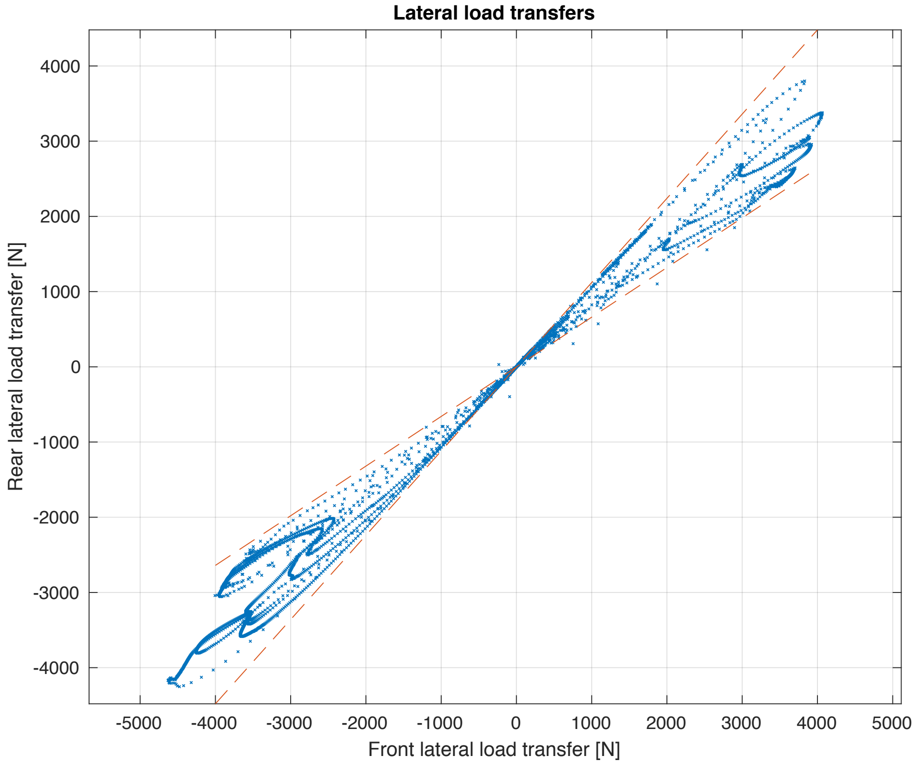


Figure 4.16: Rear axle lateral load transfer $N_{lat}^{(r)} \equiv N^{(rr)} - N^{(rl)}$ versus front axle one $N_{lat}^{(f)} \equiv N^{(fr)} - N^{(fl)}$ (blue crosses). The two dashed lines correspond to the linear relationship $N_{lat}^{(r)} = 1.12 N_{lat}^{(f)}$ and $N_{lat}^{(r)} = 0.66 N_{lat}^{(f)}$. It is evident that the roll stiffness $N_{lat}^{(r)} / N_{lat}^{(f)}$ varies of a factor of almost 2 along the track.

tion is shown in figure 4.16, where the two dashed lines correspond to a constant ratio $N_{lat}^{(r)} = 1.12 N_{lat}^{(f)}$ and $N_{lat}^{(r)} = 0.66 N_{lat}^{(f)}$. It is clear that the roll stiffness $N_{lat}^{(r)}/N_{lat}^{(f)}$ changes by a factor of almost 2 along the track, depending on the trim of the car. Since the roll stiffness is determined, at a first approximation, by the ratios between tyre, anti-roll bar, coil spring, torsion bar stiffness and by the suspension design, its variation is caused by the non-linear suspension rates (the non-linearity is due to the bump rubber forces) and suspension kinematics. In different words, a non constant roll stiffness means that, depending on the instant trim of the car, each wheel "sees" a different reduced vertical stiffness to the chassis, and the roll stiffness varies as a consequence. This effect instead is not captured by the most commonly used QSS car models where, in order to determine the loads on each wheel, it is generally assumed that the roll stiffness is constant. Thus this effect is neglected in works assuming a constant roll ratio as [21, 22, 23, 49, 72, 65] and [58].

4.4 Racing go-kart

In the previous sections of this chapter different models have been proposed for lap time simulation of racing cars; this section instead is focused on the modelling and simulation of go-karts. This section is motivated mainly by two reasons: first, go-karts are four wheeled vehicles with a significantly different dynamics than cars and, second, in literature there is lack of optimal control based simulations regarding go-karts.

Go-karts are four wheeled vehicle that, unlike cars, do not have suspensions nor differential, therefore their dynamics is different from that of other four wheeled vehicles. The absence of rear axle differential makes the two rear wheels to have a strictly correlated longitudinal slip and this characteristic makes it more difficult for a kart to turn than cars. Indeed, when turning at constant speed, the rear inner tyre has a positive slip while the rear outer tyre has a negative one; thus longitudinal forces that tend to oppose to cornering are generated. In order to reduce this effect, the inner rear wheel should be unloaded as much as possible: it's quite usual to see expert drivers lifting up the such wheel while cornering. On the contrary, rear-driven race cars tend to lift the inner front wheel when cornering. The frame stiffness “seen” by each wheel and the steering geometry are the most important parameters that affect this phenomenon; moreover, the absence of suspensions makes the tyre and the frame compliances absorb all shocks induced by road unevenness. Thus kart frame stiffness becomes a key feature that profoundly affects kart performance, as highlighted in [69] through a multibody simulated analysis and a comparison with experimental data; such study was focused on circular trajectories at constant speed only.

In this section kart dynamics has been studied with use of lap time optimal control simulations. Simulations have been carried out on the track “Pista Azzurra”, in Jesolo (IT), and the results have been validated by comparison with experimental data acquired on the same track. In the following subsections the mathematical model of the kart and the optimal control formulation are described. Then the model validation is presented, including the kart analysis in laboratory, the data acquiring and the telemetry-simulation comparison. Finally, the simulations outcomes are presented and particular attention is dedicated to the go-kart dynamics and tyre slippage along a turn.

4.4.1 Go-Kart Model

A go-kart is a four wheeled, rear axle traction vehicle whose distinguishing features are the absence of the suspensions system and differential. Since the vehicle has four contact points with the road, it is hyperstatic. Therefore, the load distribution on each tyre depends on the tyre radial compliance as well as the compliance of the chassis, which is quite deformable.

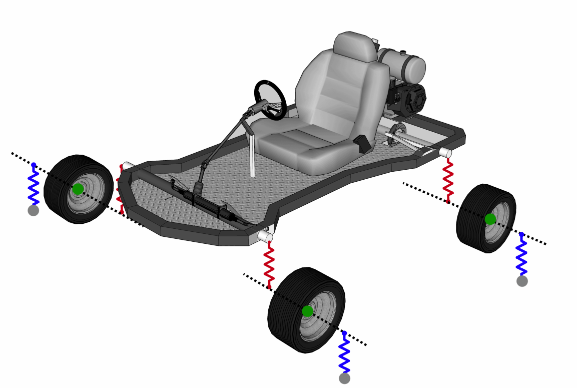


Figure 4.17: Schematic go-kart model. Red springs, that connect each wheel to the chassis, represent the chassis vertical stiffness seen by the wheel. Blue springs, connecting each wheel to the asphalt (grey circles), represent the tyre radial stiffness. Anti-roll bars (not represented) connect the left and right wheel joint points (green circles).

The absence of the differential on the traction axle makes any cornering manoeuvre problematic. Indeed, while cornering, the inner rear wheel has a smaller forward velocity, but the same spin velocity than the outer one. Thus the inner longitudinal slip is bigger than the outer one, which leads to a yaw torque opposite to the yaw rate. Since the longitudinal force is approximately proportional both to the longitudinal slip and tyre load, to reduce such undesired effect, the vertical load of the inner wheel should be as low as possible and ideally null. This condition may be achieved by designing a proper combination of tyres and chassis stiffness, as well as steering system geometry. Steering geometry indeed plays an important role in load transfers: due to the particular linkages between front wheels and chassis, when steering the inner front tyre gets pulled down by a quantity proportional to the steering angle, while the outer front tyre is lifted up by the same amount. Thus, part of the vertical load is transferred to the the front inner wheel and the rear inner one gets more - if not completely - unloaded. Such essential features are fully embraced in the mathematical model of the vehicle here developed. The inertia of the vehicle (chassis, engine, tanks, steering, wheels, etc.) is modelled with a single rigid body. Indeed vehicle vibrations and any high frequency dynamics are not of interest in lap time simulations. The mass and inertia of the rider are also incorporated in such rigid body, since the seat is very stiff and the rider body is constricted. Some racing riders are still able to partially move their torso while cornering; however, this effect is not so relevant and it is completely neglected here. To capture the actual distribution of vertical tyres load, the chassis compliance seen by each wheel is modelled with a lumped parameters approach using four vertical linear springs that connect each wheel hubs to the chassis, while other four vertical linear springs connect wheel hubs to the ground and represent tyres radial stiffness, as depicted in figure 4.17. Front and rear anti-roll bars, not shown in the figure, are included; as it will described in the following sections, they exert to each wheel a force proportional to the difference between the right and left wheel travels. The 125cc go-kart studied in this article was equipped with rear brake only, so the spin dynamics of the rear axle is included into the model, while front axle is neglected. According to this description, the vehicle model has a total of seven degrees of freedom (dof), six of them associated to the vehicle gross motion and the latter associated to the rear axle spin rotation.

The road model and the vehicle tracking variables used here are the same of the one described in section 4.2.1 and 4.3.1. The $2D$ road model is here adopted because the "Pista Azzura" race circuit is flat. As in section 4.2.1 and 4.3.1, n, α and z represent respectively the chassis lateral displacement from the road middle lane, the chassis relative heading angle and the CoM vertical displacement from the nominal height h. Moreover ϕ and μ are the chassis roll and pitch angles, and, since the chassis and tyres are very stiff, they are considered to be small, i.e. all functions of ϕ and μ are expanded in Taylor series up to the first order in such variables. Finally, V, λ and Ω are respectively the speed, drift angle and yaw rate of the $\boldsymbol{W}_{v0}$ frame (4.3), as in the car model described in section 4.2.1.

Newton's equations, that describe the vehicle translation respectively along the longitudinal, lateral and vertical directions, are similar to the ones obtained for the 14 GT car model (4.4). The differences are the absence of the inertial terms due to the suspensions, the absence of the front tyre longitudinal forces, moreover the steering δ and sideslip λ angles are not assumed to be small since go-karts may be driven at relatively high slip and steering angles. Thus the resulting Newton equations are:

$$m\left(\frac{d(V\cos\lambda)}{dt}+\Omega V\sin\lambda\right)=F_{tx}^{(rr)}+F_{tx}^{(rl)}-\left(F_{ty}^{(fr)}+F_{ty}^{(fl)}\right)\sin\delta-F_d \tag{4.63a}$$

$$m\left(\Omega V\cos\lambda-\frac{d(V\sin\lambda)}{dt}\right)=F_{ty}^{(rl)}+F_{ty}^{(rr)}+\left(F_{ty}^{(fl)}+F_{ty}^{(fr)}\right)\cos\delta \tag{4.63b}$$

$$m(\ddot{z}-g)=-\sum_{\hat{a}\hat{s}}N^{(\hat{a}\hat{s})} \tag{4.63c}$$

where $N^{(\hat{a}\hat{s})}, F_{ty}^{(\hat{a}\hat{s})}, F_{tx}^{(\hat{a}\hat{s})}$ are respectively the vertical, lateral and longitudinal tyre forces. The front steering angle δ is assumed to be equal for both wheels.

The Euler equations of motion, calculated with respect to the origin of the frame $\boldsymbol{W}_{v0}$

defined in (4.3), are:

$$
\begin{split}
I_{xx}\ddot{\phi} + \left(I_{zz}\mu - I_{xx}\mu - I_{xz}\right)\dot{\Omega} + (I_{zz} - I_{xx} - I_{yy})\Omega\dot{\mu} \\
-M\left(h\frac{d(V\sin\lambda)}{dt} + (z-h)\Omega V\cos\lambda\right) \\
-(I_{yy} - I_{zz})\phi\Omega^2 + I_w^{(f)}\frac{d}{dt}\left(\sin\delta\left(\omega_w^{(fr)} + \omega_w^{(fl)}\right)\right) + \Omega\left(I_w^{(f)}(\omega_w^{(fr)} + \omega_w^{(fl)})\cos\delta + I_a\omega_a\right) = \tau_x
\end{split}
\tag{4.64a}
$$

$$
\begin{split}
I_{yy}\ddot{\mu} + (I_{yy} - I_{zz})\phi\dot{\Omega} + (I_{xx} + I_{yy} - I_{zz})\Omega\dot{\phi} + M\left((z-h)\frac{d(V\cos\lambda)}{dt} - h\Omega V\sin\lambda\right) \\
-\left(I_w^{(f)}\frac{d}{dt}\left(\cos\delta\left(\omega_w^{(fr)} + \omega_w^{(fl)}\right)\right) + I_a\dot{\omega}_a\right) + \left((I_{zz} - I_{xx})\mu - I_{xz}\right)\Omega^2 \\
+I_w^{(f)}\Omega\frac{d}{dt}\left(\sin\delta\left(\dot{\omega}_w^{(fr)} + \dot{\omega}_w^{(fl)}\right)\right) = \tau_y
\end{split}
\tag{4.64b}
$$

$$
\begin{split}
(I_{zz} + 2I_{xz}\mu)\dot{\Omega} - \sum_{\hat{a}\hat{s}} I_w^{(\hat{a})}\omega_w^{(\hat{a}\hat{s})}\dot{\phi} - I_{xz}(\ddot{\phi} - 2\Omega\dot{\mu}) \\
-\frac{d}{dt}\left(\phi\left(I_w^{(f)}\left(\cos\delta\left(\omega_w^{(fr)} + \omega_w^{(fl)}\right)\right) + I_a\dot{\omega}_a\right)\right) + \frac{d}{dt}\left(\mu I_w^{(f)}\left(\sin\delta\left(\omega_w^{(fr)} + \omega_w^{(fl)}\right)\right)\right) = \tau_z
\end{split}
\tag{4.64c}
$$

where $t_v^{(r)}$ and $t_v^{(r)}$ are respectively the rear and front kart half width, ω_a is the axle spin velocity, and the inertia I_a includes all parts connected to the shaft (wheels, disk brake, etc.) as well as the equivalent inertia of the engine, which is connected to the rear axle by a fixed ratio chain transmission.

$$
\begin{split}
\tau_x =& \sum_{\hat{a}\hat{s}} N^{(\hat{a}\hat{s})}(t_v^{(\hat{a}\hat{s})}) + (N^{(rl)} + N^{(rr)})h\phi \\
&+ (N^{(fl)} + N^{(fr)})\left(r_t^{(f)}\mu\sin\delta\cos\delta + \phi\left(h - r_t^{(f)} + r_t^{(f)}\cos^2\delta\right)\right)
\end{split}
\tag{4.65a}
$$

$$
\begin{split}
\tau_y =& F_l x_A - F_d y_A + (N^{(rl)} + N^{(rr)})\left(b + (h - r_t^{(r)})\mu\right) \\
&+ (N^{(fl)} + N^{(fr)})\left(a + r_t^{(f)}\phi\sin\delta\cos\delta + \mu\left(h - r_t^{(f)}\cos^2\delta\right)\right)
\end{split}
\tag{4.65b}
$$

$$\begin{aligned}\tau_z =&F_{tx}^{(rl)}(t_v^{(r)} + h\phi) + F_{tx}^{(rr)}(-t_v^{(r)} + h\phi)(F_{ty}^{(rl)} + F_{ty}^{(rr)})\left(b + \mu(h - r_t)\right)\\&(F_{ty}^{(fl)} + F_{ty}^{(fr)})\left(-a\cos\delta + \mu\cos\delta(h - r_t) - \phi\sin\delta(h - r_t)\right)\\&- (F_{ty}^{(fl)} + F_{ty}^{(fr)})t_v^{(f)}\sin\delta\end{aligned} \tag{4.65c}$$

Finally, the model is completed by the following rear axle spin equation:

$$I_a\dot{\omega}_a = \tau_d - F_{tx}^{(rr)}(r_{tr}^{(r)} - \xi^{(rr)}) - F_{tx}^{(rl)}(r_{tr}^{(r)} - \xi^{(rl)}) \tag{4.66}$$

where $r_{tr}^{(r)}$ is the rear tyre rolling radius, $\xi^{(rr)}$ and $\xi^{(rl)}$ are rear right and rear left tyre deformations, and τ_d is the axle torque due either to the engine ($\tau_d > 0$) or the brake ($\tau_d < 0$).

Tyres

Since the vehicle has four contact points with the road, it is not possible to determine the tyre vertical load unless the compliances of tyres and chassis are considered. Since in a kart the tyre stiffness is generally three to five times greater than chassis stiffness, while the wheel mass is of the order of 10^{-2} times the vehicle mass (rider included), vibration frequencies of the wheels with respect to the chassis are far faster than the eigen-frequencies of vehicle gross motion, so chassis and wheels may be considered as a unique rigid body, which is attached to the road by means of massless springs and dampers. According to these assumptions, for each wheel the load force due to tyre deformation can be considered equal to the vertical force due to chassis deformation; this approach is the same adopted in the development of the massless wheel car model in section 4.2.2. Moreover the results obtained from the comparison of the full model with the massless one justify the modelling choice to neglect wheel hop dynamics (see 4.2.6).

Following the same approach used for the massless wheel car model, chassis vertical displacements $z_c^{(\hat{a}\hat{s})}$ on each wheel attach point can be expressed as a function of the other state variables by equating the tyre load to the force due to the chassis deformation. The procedure is exactly the same that has been done for the massless wheel car model in section 4.2.2, if the suspension travel are replaced by the chassis deformations[8]. The only difference with the massless wheel model is the presence in equation (4.68) of the terms proportional to β_δ. Neglecting the damping contribute to the tyre loads, the balance equations for the

[8] From a mathematical modelling point of view, suspension travels and chassis-wheel attach point displacements are equivalent, yet their physical origin is different.

vertical forces acting on the wheels are:

$$\begin{aligned}
K_{cs}^{(r)} z_c^{(rr)} + K_a^{(r)}(z_c^{(rr)} - z_c^{(rl)}) + K_{tr}^{(r)} \xi^{(fr)} &= 0 \\
K_{cs}^{(r)} z_c^{(rl)} - K_a^{(r)}(z_c^{(rr)} - z_c^{(rl)}) + K_{tr}^{(r)} \xi^{(fl)} &= 0 \\
K_{cs}^{(f)} z_c^{(fr)} + K_a^{(f)}(z_c^{(fr)} - z_c^{(fl)}) + K_{tr}^{(f)} \xi^{(rr)} &= 0 \\
K_{cs}^{(f)} z_c^{(fl)} - K_a^{(f)}(z_c^{(fr)} - z_c^{(fl)}) + K_{tr}^{(f)} \xi^{(rl)} &= 0
\end{aligned} \tag{4.67}$$

where $z_c^{(\hat{a}\hat{s})}$ are frame vertical displacements on the wheel attach points, $K_{cs}^{(f)}$, $K_{cs}^{(r)}$ are the chassis stiffness at the front and rear axles, and $K_a^{(r)}$ and $K_a^{(f)}$ are the rear and front anti-roll bar stiffness. In equations (4.67) the first terms represent the force due to chassis deformations, the second terms are the force of the anti-roll bars and the last terms stand for tyres deformations loads. Tyre deformations can be calculated with the following expressions:

$$\begin{aligned}
\xi^{(fr)} &= z + z_c^{(fr)} - a\mu + t_v^{(f)}\phi + \beta_\delta \delta \\
\xi^{(fl)} &= z + z_c^{(fl)} - a\mu - t_v^{(f)}\phi - \beta_\delta \delta \\
\xi^{(rr)} &= z + z_c^{(rr)} + b\mu + t_v^{(r)}\phi \\
\xi^{(rl)} &= z + z_c^{(rl)} + b\mu - t_v^{(r)}\phi
\end{aligned} \tag{4.68}$$

where $\pm\beta_\delta\delta$ is the linear approximation of the kinematic relationship that link the vertical motion of the wheel centre to the steering angle. In different words β_δ is the ratio between the front right wheel vertical displacement and the steering angle. It can be noted that equations (4.68) are very similar to (4.10), with the difference that suspensions travel are replaced by chassis deformations and with the addition of the term $\beta_\delta\delta$.

Once the expressions for $z_c^{(\hat{a}\hat{s})}$ are obtained solving (4.67), they can be substituted into the tyre deformation expressions (4.68) obtaining an expressions for the tyre deformations that is similar to the one obtained for the massless wheel model ((4.24)):

$$\begin{aligned}
\hat{\xi}^{(rr)} &= K_{es}^{(r)}(\mu b + \phi t_v^{(r)} + z) + K_{ea}^{(r)} t_v^{(r)} \phi \\
\hat{\xi}^{(rl)} &= K_{es}^{(r)}(\mu b - \phi t_v^{(r)} + z) - K_{ea}^{(r)} t_v^{(r)} \phi \\
\hat{\xi}^{(fr)} &= K_{es}^{(f)}(\beta_\delta \delta - \mu a + \phi t_v^{(f)} + z) + K_{ea}^{(f)}(\beta_\delta \delta + \phi t_v^{(f)}) \\
\hat{\xi}^{(fl)} &= K_{es}^{(f)}(\beta_\delta \delta - \mu a - \phi t_v^{(f)} + z) - K_{ea}^{(f)}(\beta_\delta \delta + \phi t_v^{(f)})
\end{aligned} \tag{4.69}$$

where K_{es} and K_{ea} are the equivalent spring and anti-roll bar stiffnesses, and they are given by expression (4.24), with the only difference that suspension stiffness $K_{ss}^{(\hat{a})}$ is now replaced by the chassis stiffness $K_{cs}^{(\hat{a})}$. Finally, tyre loads are calculated using the tyre deformations $\xi^{(\hat{a}\hat{s})}$ as in (4.25).

Table 4.7: Pacejka's tyre magic formula coefficients

coefficient	rear tyre	front tyre
p_{Cx1}	2.3	-
p_{Dx1}	0.9	-
p_{Ex1}	0.95	-
p_{Kx1}	20	-
p_{Kx2}	1	-
p_{Kx3}	-0.5	-
p_{Cy1}	2.3	2.13
p_{Dy1}	1.5	1.5
p_{Ey1}	0.9	0.8
p_{Ky1}	-37.6	-34.1
p_{Ky2}	1.6	1.6
p_{rBx1}	14	-
p_{rBy1}	12	-
p_{rCy1}	0.6	-
λ_{Fz0}	1.6	1.6

All Pacejka's coefficients ([70], chapter 4) that are not listed here have been set to their neutral value (either 0 or 1).

Lateral and longitudinal tyres forces are calculated according to the Pacejka Magic Formula [70] as a function of tyre vertical load N, sideslip angle λ_t and longitudinal slip κ_t, exactly as done in section 4.2.1. The expressions for the tyre sideslip angles and longitudinal slips can be found respectively in equations (4.14) and (4.15). Moreover, relaxation equations are used on the tyre forces, as in (4.16) and (4.17)[9]. All the coefficients used in these the Pacejka formula are listed in table 4.7; the parameters that does not appear in such table have been set to their neutral value (either zero or one). This because only the tyres lateral stiffness has been measured in lab tests (as it will be described in section 4.4.3.1), thus only the smallest number of parameters in the Pacejka's magic formulas has been used.

4.4.2 State space and OCP formulation

Newton's equations (4.63) and (4.64) may be immediately reduced to the first order by introducing auxiliary variables for the relevant chassis speeds:

$$\dot{z} = z_{dot} \qquad \dot{\phi} = \phi_{dot} \qquad \dot{\mu} = \mu_{dot} \tag{4.70}$$

[9] The only small difference with the equations of section 4.2.1 is that now the angle δ is not considered small, thus expressions are not linearised in δ.

At this point, equations (4.63)-(4.66), (4.70), together with the tyre equations (4.16), (4.17), the road equations (4.31) and their subordinate expressions completely describe the go-kart dynamics as a system of 17 first order differential equations with as many state variables and 2 inputs, respectively the rear axle torque τ_d, which (mainly) control the longitudinal dynamics, and the steering angle δ, which (mainly) control the lateral dynamics.

As it has been done for the previous models, the steering angle and driving torque are not controlled directly, but via its time derivative, so as to limit jerky manoeuvres:

$$j_\delta = \dot{\delta} \qquad j_{\mathcal{T}} = \dot{\tau}_d \tag{4.71}$$

In conclusion, the vehicle dynamics is described by means of a set of two inputs:

$$\boldsymbol{u} = \begin{bmatrix} j_\delta; & j_{\mathcal{T}} \end{bmatrix} \tag{4.72}$$

plus 19 state variables:

$$\boldsymbol{x} = \left[s; n; \alpha; z; \phi; \mu; u; v; z_{dot}; \Omega; \phi_{dot}; \mu_{dot}; \omega_a; F_{ty}^{(rr)}; F_{ty}^{(rl)}; F_{ty}^{(fr)}; F_{ty}^{(fl)}; \delta, \tau_d \right] \tag{4.73}$$

and as many implicit first order differential equations.

Finally, the optimal control problem is formulated as described in section 4.2.5. The time to space transformation described in appendix A is used so that the variable s becomes the independent variable. Some constraints are added to the OCP so as to force the vehicle to stay within the track borders, not to exceed the engine maximum power and not to lock the wheels. The model parameters used for the simulations are reported in table 4.8.

The optimal control problem is solved using *Pins* (see section 3.2.1).

4.4.3 Experimental test and model validation

In order to provide a validation of the mathematical model by comparison between simulation results with experimental telemetry, a precise measurement of the go-kart characteristic is first necessary. Indeed, the reliability of the simulation is highly affected by the agreement of the data used with real kart properties.

In the next part of this section, the description of the procedure used to measure go-kart characteristics is reported, while in the second one simulation results are compared to experimental data.

Table 4.8: Go-kart dataset.

symbol	value	units	description
h	0.250	m	[m] centre of gravity (CoG) height
a	0.645	m	[m] distance between the front axle and the vehicle CoG
b	0.400	m	[m] distance between the rear axle and the vehicle CoG
p	1.045		[m] wheelbase
$2t_v^{(f)}$	1.055	m	[m] front track
$2t_v^{(r)}$	1.200	m	[m] rear track
β_δ	0.058	m/rad	[m] front wheel hub displacement to steering angle ratio
m	165	kg	[m] vehicle mass (rider included)
I_{xx}	20	kgm^2	[e] roll moment of inertia
I_{yy}	15	kgm^2	[e] pitch moment of inertia
I_{zz}	25	kgm^2	[e] yaw moment of inertia
I_{xz}	5	kgm^2	[e] mixed moment of inertia
I_a	0.2	kgm^2	[e] spin inertia of the rear axle
ρ	1.2	kg/m^3	air density
F_d	0.7	m^2	[e] drag surface coefficient
$K_{cs}^{(r)}$	60×10^3	N/m	[m] rear chassis stiffness (vertical displacements)
$K_{cs}^{(f)}$	17.7×10^3	N/m	[m] front chassis stiffness (vertical displacements)
$K_a^{(r)}$	0	N/m	rear anti-roll bar stiffness[(1)]
$K_a^{(f)}$	0	N/m	front anti-roll stiffness[(1)]
$K_{tr}^{(r)}$	61.3×10^3	N/m	[m] rear tyres radial stiffness
$K_{tr}^{(f)}$	64.5×10^3	N/m	[m] front tyres radial stiffness
$K_{td}^{(r)}$	1.0×10^3	Ns/m	[e] rear tyres radial damping
$K_{td}^{(f)}$	1.0×10^3	Ns/m	[e] front tyres radial damping
$r_{tr}^{(r)}$	0.139	m	[m] rear tyres rolling radius
$\tau_e^{(max)}$	17.6	Nm	[m] maximum engine torque (@ 10250 rpm)
$P^{(max)}$	20.1	kW	[m] maximum engine power (@ 11500 rpm)

[m]: Measured value, [e] :Estimated value, [f]: Fitted value.

[(1)]: The go-kart used for the track test was not endowed with anti-roll bars.

4.4.3.1 Lab tests

The geometry, inertia and compliance of the go-kart, as well as tyre properties, have been measured to feed the simulation model. The geometry of the chassis have been carefully measured by using a 3D coordinate-measuring machine. Weighing balances have been used to measure the go-kart mass and its longitudinal position, the CoG vertical position was estimated by tilting the vehicle until it reached its (unstable) equilibrium position on the two lateral wheels. Such geometric and inertial data was used to build a virtual prototype of the go-kart, which has been used to estimate the moment of inertia as well as to estimate the frame stiffness by means of a FEM analysis. The frame is complex and its representation in term of few lumped springs used in the vehicle dynamic illustrated section 4.4.1 disregards many degrees of freedom and the correlation of such lumped stiffness to FEM analysis is not trivial. As previously described, in this case the lumped stiffness are four linear springs that connect each wheel centre to the chassis, as depicted in figure 4.17. Such stiffnesses have been estimated by simultaneously locking three wheels hub and by applying a vertical force on the last one. Vehicle parameters are collected in table 4.8.

Tyres properties have been measured on the rotating tyre test rig which is available at the Department [26]. The identification of the vertical structural stiffness $K_{es}^{(r)}$, $K_{es}^{(f)}$ has been carried out by applying different vertical loads in the range of $0-1500N$ both front and rear tyres showed a good linear behaviour. Tyre adherence was measured with a vertical load of $560N$, which is approximately the load of a rear tyre in static conditions. Figure 4.18 shows the ratio between the lateral force and vertical load as a function of the sideslip angle, for both the front and rear tyre. The tyre behaviour in the range between $0°$ and $6°$ of sideslip angle has been accurately identified, but unfortunately the adherence peak was not reached during measurements; indeed, the friction between the tyre and the test rig were exaggeratedly high (as the tyre test rig was not specifically designed to measure kart tyres). It is worth pointing out that in actual driving conditions the adherence limit remarkably depends on many parameters and in particular on the asphalt characteristics. Hence the correlation with lab measurement would not be very easy in any case. On the contrary, the cornering stiffness, i.e. the tyre behaviour at low sideslip angles, mainly depends on the tyre carcass characteristics and it is not so much influenced by the tyre-road friction properties [70] [42]. For this reason, tyres adherence peak have been adjusted to fit the experimental maximum accelerations (that can be read in figure 4.21). Tyre forces that were used for the simulation are reported in figure 4.19. Rear tyre normalized lateral force is shown for different values of longitudinal slip to highlight the coupling between lateral and longitudinal forces. For the front tyre the force is shown only for $\kappa_t = 0$ because the front tyre's longitudinal slip is not modelled. Tyres adherence values, together with all coefficients

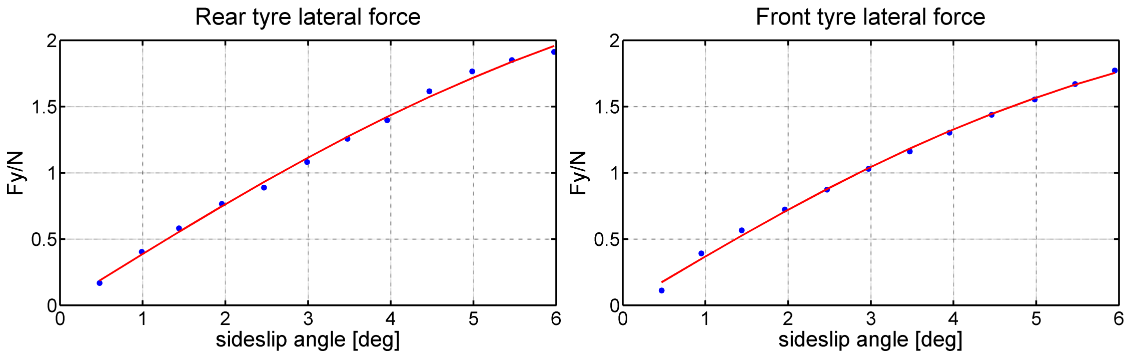


Figure 4.18: Lateral force of front and rear tyres measured on the test rig. Continuous red lines represent data fit.

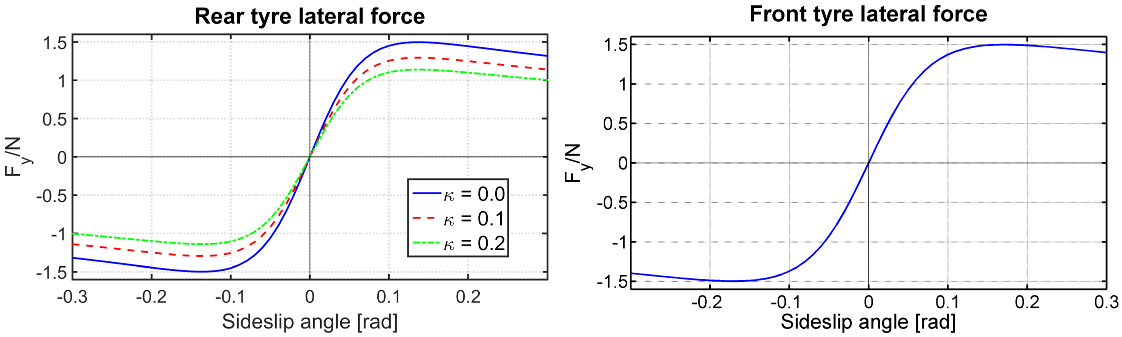


Figure 4.19: Tyre forces: normalized lateral forces are shown for rear and front tyre at a vertical load of $560N$. For rear tyre the force is shown at different values of longitudinal slip.

of Pacejka's magic formula, are summarized in table 4.8. The lateral friction peak coefficient (1.50) is significantly higher than longitudinal one (0.90) because in kart tracks thin layers of rubber get often deposited on the asphalt in curves due to high tyre slippage. Thus turns are characterized by higher friction coefficient than straights, and this effect can be modelled by considering tyres with greater lateral adherence. Such adherence conditions are confirmed in section 4.4.3.3 by the validation of the model with experimental data. The engine torque curve used in this simulation is the one provide by Iame (`www.iame.it`).

4.4.3.2 Track tests

The go-kart has been equipped with an Inertial Measurement Platform (IMU) composed by three accelerometers and three gyrometers. Moreover, a Hall sensor has been installed on the rear axle to measure the spin velocity and a rotational potentiometer has been installed on the steering system to measure the steering angle. All sensors have been connected to

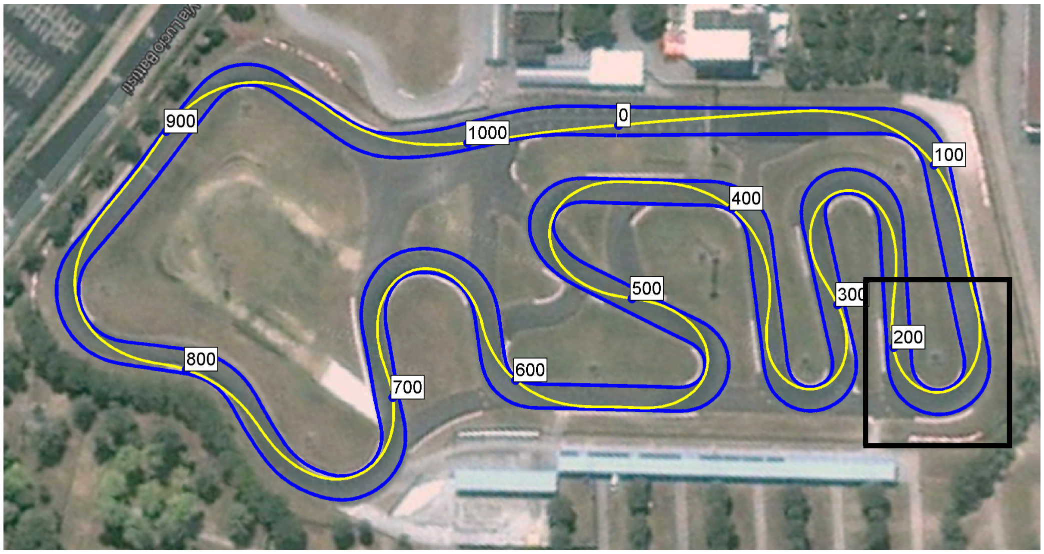


Figure 4.20: Satellite view of the “Pista Azzurra” with simulation trajectory overlying (yellow line). Numbers inside boxes shows the value of the curvilinear abscissa s along the track every $100m$. The turn inside the black rectangle is the one analysed in the next chapter.

the data logger by means of a dedicated CAN bus. Tests have been carried out at “Pista Azzurra” (Jesolo), which is a national category race-track $1051m$ long with an average width of $8m$. Test were carried out on a cold day in November, with adherence conditions far from being optimal, by an expert driver who participates in go-kart races at Italian level. Figure 4.21b and 4.21c highlighted the bounds of both the lateral accelerations $|a_y| < 15m/s^2$ and longitudinal one $-7m/s^2 < a_x < 5m/s^2$. While the traction acceleration is limited by the engine power, lateral and braking accelerations are limited by tyre adherence, where lateral adherence is much bigger than the longitudinal one thanks to the rubber deposits that tyres leave in curve. It is worth pointing out that, most likely, the driver was not able to drive very close to the tyre adherence limit and hence the acceleration bounds are a measure of driver’s capability rather than tyre adherence. However, this is not a big issue for the validation of the simulation software, as the focus will be posed on the comparison between speed profile of real and virtual driver, constrained within the same accelerations bounds.

4.4.3.3 Validation

The simulation took approximately $230s$ on a common laptop equipped with an *Intel Core-i7* 640M processor. Figure 4.20 shows a satellite view of the track with the overlying trajectory resulting from optimal manoeuvring method. The trajectory confirms that the kart remains

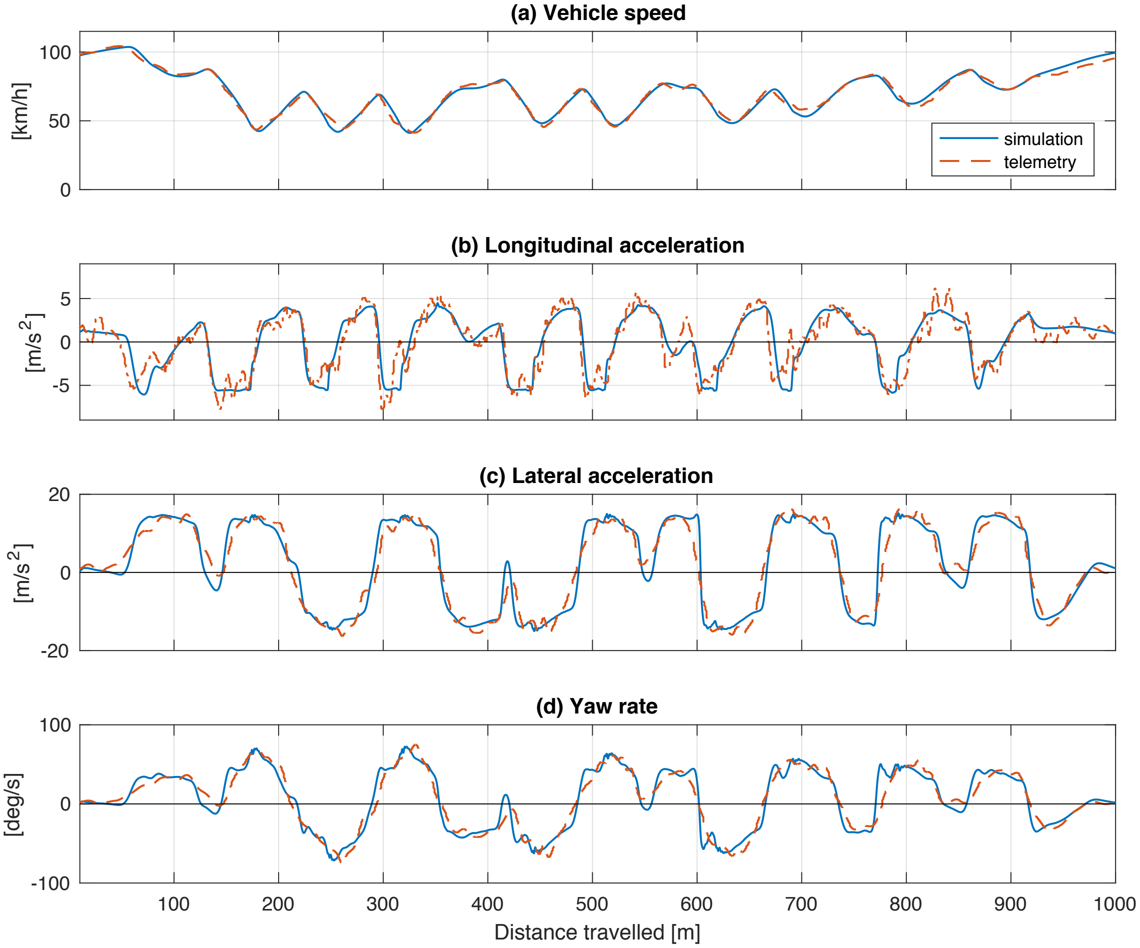


Figure 4.21: Vehicle gross motion for the full lap: dashed red lines refer to telemetry data, continuous blue ones to simulation results.

always within borders of the real track, suggesting that road has been accurately reproduced. Simulated and experimental time-lap are respectively $53.575s$ and $53.580s$. However, this is not an indication of the quality of the simulation since the information on the time lap has been used to fine tune tyre adherence, which has been estimated from measured lateral and longitudinal accelerations. The accuracy of the simulation should be instead evaluated by comparing measured data with simulated ones as they both vary along the track, as reported in figure 4.21: dashed red lines refer to telemetry data, continuous blue ones to simulation. The simulated speed profile reproduces faithfully the experimental one; there are some slightly discrepancies in the speed at the middle of some turns, where the simulation is sometime faster, other slower. In addition to the simulation errors that are certainly present because of model approximation and parameter inaccuracy, it should be kept into consideration that asphalt was not perfectly dry so some parts of the circuit had a different adherence compared to others. Moreover, the driver was very skilled but not professional. So, it is opinion of the author that the accordance between simulation and experiments is excellent. Longitudinal and lateral accelerations also reproduce faithfully the measured trend. Simulated signals are obviously smoother than the experimental ones, due to the absence of every source of unwanted vibrations. For this reason, the peak value of experimental accelerations are higher than the simulated one. The experimental lateral acceleration and yaw rate signal are less affected by external noise with respect to longitudinal acceleration and they have a better correspondence with the corresponding simulated quantities.

Figure 4.22 shows additional variables that are available for the simulation only and that are quite useful to analyse the go-kart behaviour. Rear tyre longitudinal and lateral slips are reported respectively on the first and second line. As shown in the figure, sideslip angles of rear tyres are quite identical for left and right sides, while for the longitudinal slips there are significant differences. Due to the absence of a differential, such slippage differences are consequences of the yaw motion only. Thus, at a given time only one tyre can have the combination of κ_t and λ_t that produces the maximum force: this means that it is not possible to bring both tyres at adherence limits in the same manoeuvre, and the optimal control obviously tends to engage to the limit (at a longitudinal slip κ_t of ≈ 0.105) the tyre that has more load, i.e. the outer one. When it reaches such conditions, the inner one consequently is forced to have a greater slippage and reaches the high slip values (≈ 0.17) that are beyond the maximum of the Pacejka's magic formula. Regarding sideslip angles, tyres are never pushed beyond the maximum of the magic formula thanks to the optimality condition. Moreover only the front tyre sideslip angle go slightly beyond the maximum adherence value ($\lambda_t \approx 0.17rad$). This is not in contradiction with the optimality

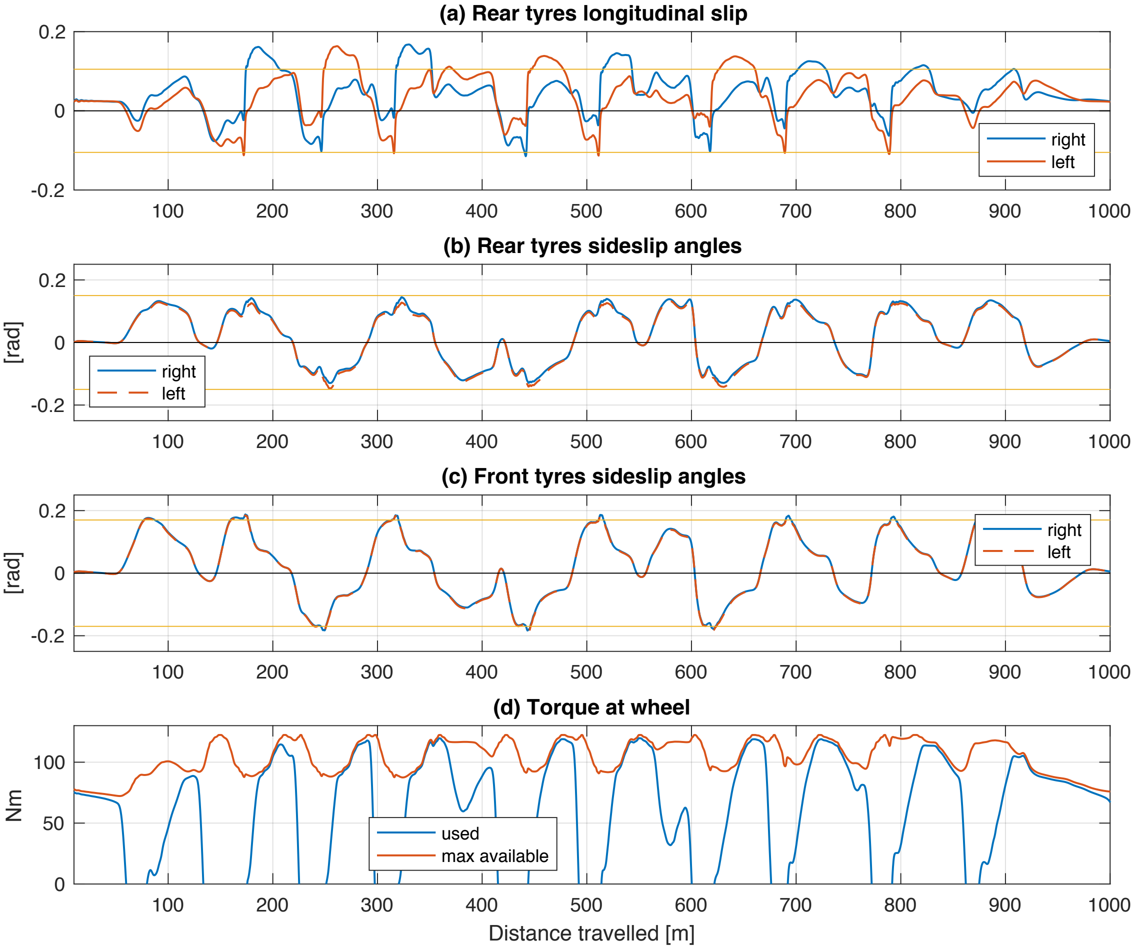


Figure 4.22: Simulation details: in tyre slip charts, blue lines refer to the right tyre, red lines to the left one. Horizontal yellow lines represent the value at which the maximum of tyre adherence is reached for a non-combined tyre force. In the chart *d*, the blue line represents the used torque, the red one the maximum available torque.

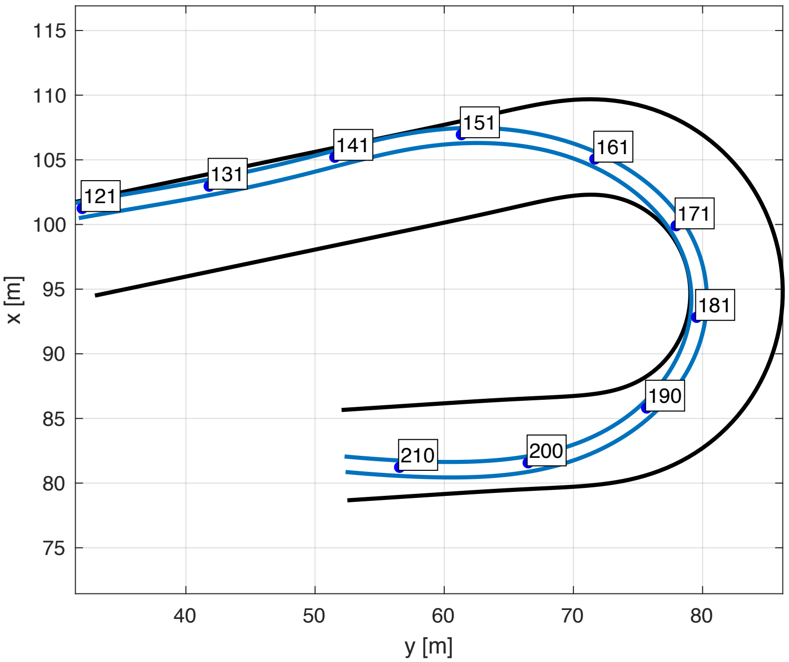


Figure 4.23: Turn analysis: optimal trajectory. Blue lines represent kart edge trajectories, numbers in rectangular box show the value of the curvilinear abscissa s. The location of the turn in the circuit is shown in figure 4.20

condition because lateral force does not decrease suddenly after the peak (see figure 4.19); moreover, keeping the front tyres at a higher steering angle helps the vehicle to brake. The rear tyres instead are limited to lower sideslip angles because their lateral slippage is due only to kart drift angle. However, thanks to the higher sideslip stiffness, the rear tyres get extremely close to their maximum lateral adherence (at $\lambda_t \approx 0.15 rad$). Finally, the last graph reported in figure 4.22 shows that engine torque represents the limiting factor when traction is required, and the delivered thrust never exceeds the maximum available power.

4.4.4 U-turn analysis

In order to better understand kart dynamics, this section focuses on in-depth analysis of the vehicle behaviour while running along a particular curve of the entire track, the clockwise one comprised between $120m$ and $215m$ of the curvilinear abscissa s. This is the turn within black box in figure 4.20; the detail of the track and the optimal trajectory are shown in figure 4.23: the kart reaches the external track borders at the entrance of the curve, then it passes at the apex in the middle of the turn, finally, after exiting the curve, it remains in a central position before approaching the next, counter-clockwise turn. The speed profile in figure

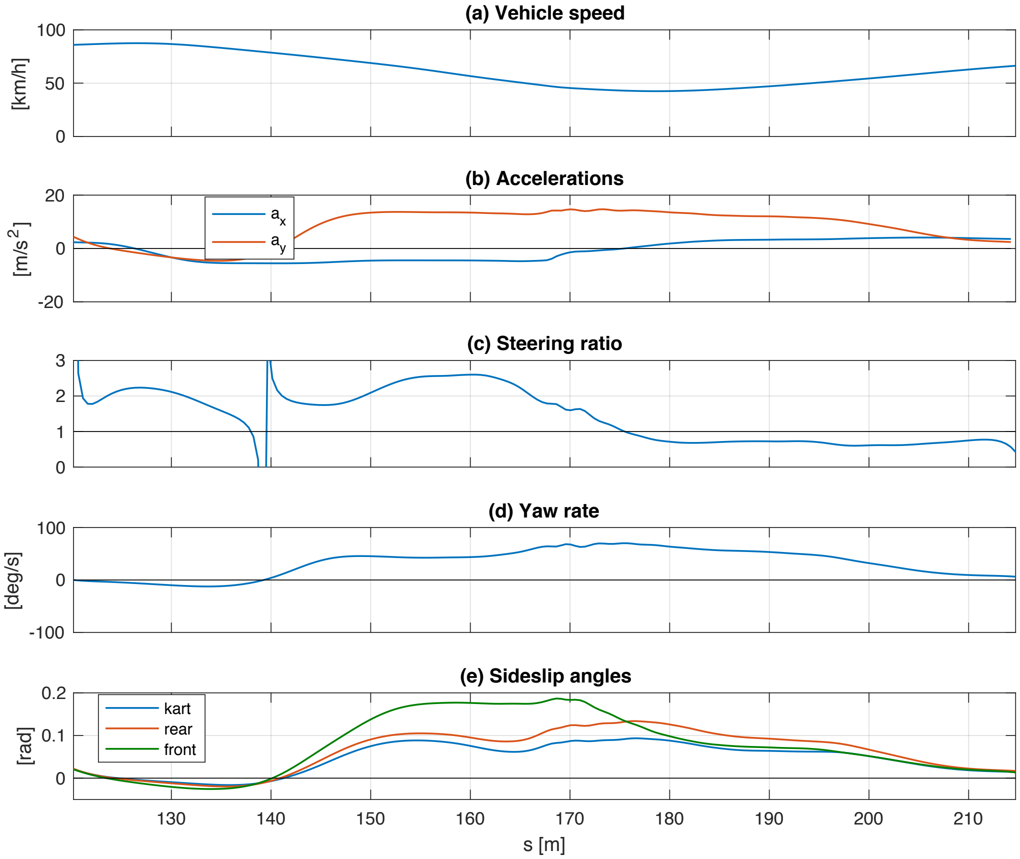


Figure 4.24: Turn analysis: gross-motion. Steering ratio shown in the chart c is the ratio between the real curvature radius and the kinematic one (that is the radius of curvature at zero slip). Values higher than 1 indicate that the vehicle is under-steering, lower than 1 over-steering. Rear (front) sideslip angle in figure e is the average of the rear (front) left and right tyres.

4.24 shows that the kart starts from a speed of $\approx 85km/h$ and brakes to $\approx 45km/h$ with a maximum longitudinal deceleration $a_x \approx -5.5m/s^2$: even if the estimated longitudinal tyre friction is equal to 0.91, the maximum deceleration is significantly lower than $0.91g$ because the kart has brakes on the rear axle only. Braking deceleration remains nearly constant to its maximum until $s \approx 165m$, then for $165m < s < 185m$, a_x grows up to its maximum value of $4m/s^2$. Such acceleration is kept nearly constant while exiting the turn. Lateral acceleration also is kept almost constant at $\approx 13m/s^2$ while turning ($150m < s < 200m$). Steering angle, drift angle and yaw rate trends highlight that a small *pendulum* manoeuvre is exploited before the beginning of the turn. Summarizing, the manoeuvre can be dividend into three phases that can be roughly associated to braking ($125m < s < 165m$), cornering ($165m < s < 185m$) and accelerating ($185m < s < 210m$). The ratio between the real curvature radius and the kinematic one (steering ratio) is lower than one after the second half of the manoeuvre, when the vehicle is accelerating, and it's greater than one during all the rest of the time except for a short transient at turn entrance in correspondence of the *pendulum* manoeuvre. When the yaw rate passes through zero the steering ratio diverges to infinity since the kinematic curvature radius is zero. The under-steering behaviour of the kart at the entrance of the turn is confirmed also by figure 4.24e, which shows that the front tyres sideslip angle is almost the double of that of rear ones.

The tyre slip and forces that are generated in this manoeuvre are shown in figure 4.25. In the braking phase, longitudinal slips are obviously negative for both tyres and they are near the value at which the maximum of the (uncoupled) longitudinal adherence is exerted. Maximum longitudinal adherence is clearly never reached because of the lateral sideslip angle that is not zero. Then, while turning, the longitudinal slip of the two tyres shows significant differences due to the yaw motion (the absence of differential forces the wheels to have the same spin) and, since the yaw is positive, the right tyre has always a higher slip. When the kart has exited from the turn and the yaw rate is nearly zero, the longitudinal slip of the two tyres becomes equal again. Regarding the lateral slippage, the sideslip angles of rear tyres starts increasing from zero when the kart enters the curve, then they reach a maximum at the middle of the turn and then decrease and become zero again when the kart has exited the turn. The rear tyre sideslip angles never reach the value at which the maximum lateral adherence would be generated. Concerning front tyres, it can be noticed that the difference between right and left load is significant but not as huge as for rear tyres. This is a clear evidence that the steering effect transfers a considerable load to the front inner wheel. Finally, the sideslip angles of front tyres have a trend that is almost proportional to that of the steering angle (figure 4.25d, cyan line).

Adherence ellipses shown in the bottom-right corner of figure 4.25 confirms what has

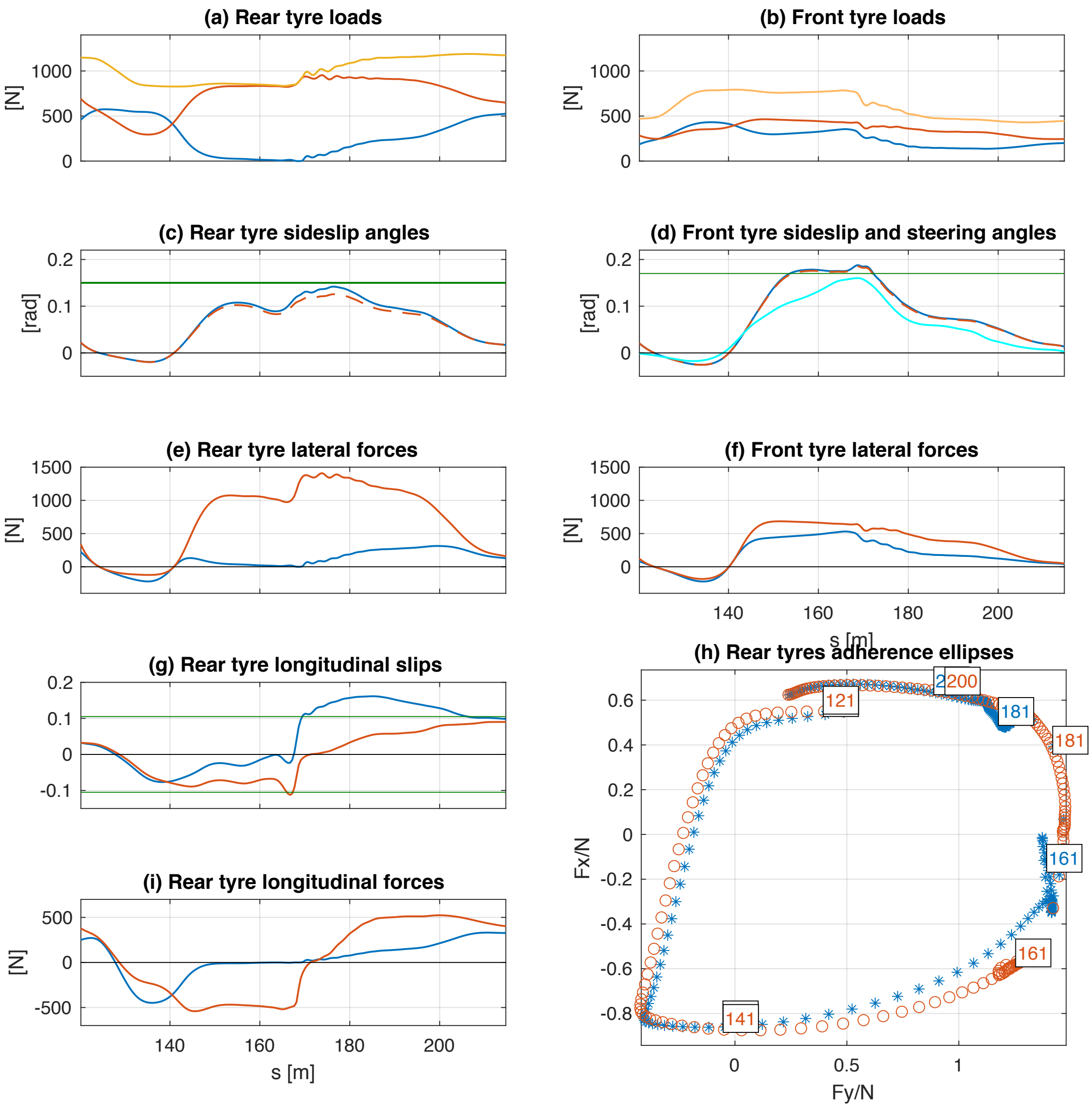


Figure 4.25: Turn analysis: tyres kinematic. Blue lines refers to right tyres, red ones to left tyres. Yellow lines, when present, show the sum of left and right tyres. Horizontal green lines in figure *c*, *d* and *g* represent the value at which the maximum of tyre adherence is reached for a non-coupled tyre force at nominal load. In plot *d* the cyan line represents the steering angle. In figure *h*, asterisks refer to right tyre, circles to left one; numbers within rectangular box show the value of the curvilinear abscissa s of the point. Turn manoeuvre begins from the top of the picture and then continues counter-clockwise. Tyre vertical loads in *a* and *b* clearly highlight how the right rear tyre (inner tyre) gets completely unload during the manoeuvre.

been said in the previous chapter: in every cornering only the outer tyre has the combination of longitudinal slip and sideslip angle engage it to the limits of adherence ellipses. Indeed red points are arranged over a bigger ellipsoidal shape. Finally, from the comparison of rear tyres longitudinal forces in figure 4.25 with the steering ratio shown in figure 4.24, it can be noticed that, at least as rough approximation, the vehicle is under-steering when braking, and over steering when accelerating. Taking into consideration the rear tyre load distribution, this behaviour is reasonable: since most of the longitudinal force is exerted by the outer tyres (the inner tyre has is less loaded, or even is fully unloaded), when braking such force generates an aligning yaw torque, and a turning yaw torque when accelerating.

4.5 Summary

The state of the art of minimum time optimal control simulations for race cars has been considerably improved in the last years thanks to the increase in computer processing power and enhancement of numerical solver; nowadays relatively complex car models can be used for optimal control based simulations. However, car multibody models for optimal control lap time simulations are still generally based on quasi steady state tyre loads in order to reduce the state-space dimension (thus the numerical size) of the resulting problem and to make the problem easier to solve. Such assumptions can lead to suboptimal results when lap time simulations are used to optimize car setup parameters related to the car dynamics, like the centre of mass position or the suspension design.

In the first part of this chapter three car models with different dynamics detail have been compared for minimum lap time problems in order to understand whether simplifying modelling assumption can affect simulation outcomes. The most accurate car model includes the chassis as a rigid body with six dof together with the wheels vertical motion and spin rotation, for a total of 14 dof. Tyre loads are calculated dynamically through tyre radial deformations. The other two models have been derived form the first one by introducing some simplifications: in the ten dof model the wheel mass is neglected, yet the chassis still includes siz dof, and in the basic seven dof model the chassis motion due to suspensions is neglected too. In the ten dof model tyre loads dynamics is still included while in the basic model quasi-steady state tyre loads are used. The aim of the ten and seven dof models is to provide simpler tools to use in optimal control simulation without losing in accuracy of the simulations outcomes. Lap time simulations have been performed on the Adria International Raceway circuit. Results showed that simulation outcomes obtained with the massless-wheel model are in agreement with the ones given by the full model: both models lead to the same[10] simulated lap-time and optimum value of the braking bias and suspension stiffness. Moreover, the massless wheel model brought to a reduction of the 43% in the computing time. Differently, the basic model highlighted remarkable differences: not only the simulated lap time are less close to the full model ones (difference of $\approx 0.4s$, but also in the two parametric analysis the best performance is obtained for different position of the CoM and different front spring stiffness than the other models. Moreover, even imposing on the basic model the exact trajectory obtained through the full one, significant differences in the tyre loads have been highlighted. Thus, despite the basic model has the lowest computing time (67% less than the full one), the loss in dynamics accuracy seems to

[10] Results given by the massless wheel and full model are not exactly the "same" since small numerical difference are unavoidably present. However, such differences are so small that they are not meaningful from a practical point of view.

affect too remarkably the simulations outcomes, at least for the CoM and spring stiffness optimisation. Among the three models, the massless wheel one gives the best compromise between short computing times and accuracy of the simulation results. Finally, it should be considered that the massless-wheel model relies on the simple force balance equation 4.22 resulting from the elimination of the wheel mass. Indeed, since equation 4.22 is linear in the suspension travel, it has been possible to express it as function of the other state variables. If a more accurate suspension model including the suspension kinematics is used (so as that the wheel does not move only along the vertical axis), then the force balance equation would result to be much more complicated and it would include also the tyre lateral and longitudinal forces through a nonlinear relationship. Thus, it would be no more possible to express the suspension travel as function of the state variables as it has been done here. However the massless wheel simplification could be anyway obtained by imposing the force balance equations as OCP path constraints (if the OCP solver allows it), leading to four additional equations (one per each suspension), instead of the eight additional equations of motion for the suspension dynamics (two first order equations per suspension).

In the second part of this chapter an improved model for GP2 cars (and other formula cars) has been developed. The aim of the GP2 model is to extend the 14 dof car model by the including the full suspension kinematics and an accurate aerodynamic forces. At the same time the model must be concise enough in order to be used for optimal control applications; the symbolic approach in deriving the equations of motion is here of great importance. More in detail, the GP2 formula car multibody model includes: chassis, suspension and wheel dynamics, full Magic Formula tyre forces, non linear tyre loads and ride-height-dependent aerodynamic forces. Suspension kinematics has been deeply analysed and included in the car model, as long as it has a significant effect on tyre loads; moreover its impact on overall car performance has been highlighted in steady state conditions. An indirect optimal control approach has then been adopted to successfully perform a full lap time simulation on the circuit of Montmelò in a reasonable amount of time (approximately half an our on a common computer), and the model has been validated by comparison with the telemetry data of an official qualifying lap in 2012. The experimental data have shown a good agreement in speed, accelerations and accelerations-speed dependence. The simulated tyre loads highlighted a roll stiffness that changes by a factor ≈ 2 along the circuit. Commonly used car models based on quasi steady state loads are not able to capture this effect as long as they assume a constant roll stiffness. This shows that the additional complexity of the car model here developed do really lead to an improvement of the simulated tyre load dynamics.

The third part of this chapter has focused on lap time simulation of go-karts. This type of four-wheeled vehicles is indeed significantly different from race cars due to the absence of suspensions and rear axle differential. The multi-body model here adopted is equivalent to the ten dof massless-wheel one, once the suspensions are replaced by the chassis stiffness at the wheel attachment points. Of course rear wheels have the same speed. This model, fed with go-kart characteristics measured in lab tests, has been demonstrated to reproduce enough accurately the telemetry data recorded during experimental track test with an expert driver. Measured speed and accelerations showed good agreement with simulated signals. Simulations results have been exploited to study the dynamics of go-karts, with particular attention to tyre slippages. Racing manoeuvres that bring to the lifting of the rear inner wheel when turning has been reproduced in the lap time simulation. The importance of such manoeuvre due to the lack of differential has been highlighted too.

Chapter 5

Design optimisation of an electric motorbike for the Tourist Trophy Zero Isle of Man competition

In this chapter optimal control lap time simulations are applied to the design optimisation of an electric motorbike taking part at the Tourist Trophy Zero race. While in the previous chapter particular attention was put in the modelling of the vehicle dynamics, the main focus is here shifted to the powertrain as long as it is the key element of electric vehicles and it will be object of an optimisation study.

5.1 Introduction

Electric (EV) and hybrid electric (HEV) vehicles are continuously gaining in popularity as the automotive industry is responding to tough environmental regulatory challenges and is aiming to meet growing consumer demand for energy consumption reduction and more sustainable living. Contemporary with the wider adoption of electric road vehicles, motor-sport too has seen an electrification movement. More driven by the innovation aspect and opportunities through technological advantage, it brought forward consideration for low carbon racing categories. The pinnacle of motor-sport, Formula 1, opened up regulations to introduce hybridization in 2009. Energy-efficiency competitions have gained widespread appeal such as the Shell Eco Marathon, while an all new FIA electric racing single seater championship Formula E saw the light in 2014. Motorcycle racing led the way however, with TTXGP holding the first international low carbon, zero-emission Grand Prix on the Isle of Man in 2009. TTXGP was succeeded by the TT Zero Challenge and incorporated as an official event class of the historic Isle of Man Tourist Trophy or TT from 2010, where racing has been taking place annually on the infamous $37.73mi$ Snaefell Mountain course since 1911.

Generally speaking, the energy consumption and performance of EVs depend significantly on driving inputs and vehicle design parameters, therefore optimisation techniques have been applied to EVs as demonstrated in literature [51, 97, 96, 87, 80, 29, 74]. For road vehicles, where the performance is not of primary importance, optimisation is mainly related to energy management, while for racing vehicles the main goal is minimum time optimisation [58, 61, 3].

TT Zero is a time trial race whereby electric motorcycles aim to complete one $60km$ lap on closed public roads along the Snaefell Mountain Course. Achieving minimum time is thus the main goal, and the optimisation of such a motorcycle would fall into the racing category. However, energy management optimisation is not negligible due to the unusual length of the TT course, therefore the TT Zero Challenge fits in-between a road and race application and corresponding design considerations must be made.

This chapter aims to optimise through optimal control lap time simulations a TT Zero motorcycle design, as run by the Brunel Racing team from Brunel University London. Brunel Racing has entered an electric racing motorcycle in the inaugural TTXGP and in every subsequent TT Zero Challenge to date. Experimental data gathered during testing and racing on the Isle of Man will serve to validate the optimal control model presented here. Using an optimal control approach for the TT Zero application is a challenging proposition for two main reasons: first, the length of the Mountain Course is nearly 12 times that of a common racing circuit; second, the mathematical model must capture both the vehicle dynamics and power-train characteristics.

Despite optimal control simulations have been widely used in recent years for lap time optimisation of various vehicles, to the best of the author's knowledge, such work has not yet been reported on for applications comparable in complexity to that of the TT Zero Challenge. Optimisation studies related to electric vehicles have thus far been performed with other strategies than optimal control simulations [79, 51, 97, 96, 80, 29], even though optimal control theory could be applied, as already demonstrated by [87, 74, 58]. The model presented here does for the first time offer a comprehensive lap time and design optimisation for TT Zero, by incorporating a motorcycle dynamics model, an electric power train configuration model and a race course over a long distance with three-dimensional track geometry including non-negligible slope and elevation changes.

Using a sound modelling approach for all relevant characteristics of motorcycle, power train and road is key to successfully solve the optimal control problem. Indeed, the complexity of the motorcycle and power train models, together with the long road that increases the numerical size of the problem, may lead to long computational solving times or even to a non-solvable problem.

In the next section, the motorcycle and power train models are described in further detail. The mathematical models introduced aim to correctly capture real world behaviour, while remaining simple enough to allow computations to run in just a few minutes. Simulation outcomes are then validated through comparison with experimental data from the TT Zero race. Finally, the models are used in a design optimisation exercise, focusing on three critical performance parameters for the electric motorcycle: direct drive gear ratio, battery pack size and motor configuration.

5.2 Road

The TT course runs for $37.73mi$ or approximately $60km$ over closed public roads and the Snaefell Mountain on the Isle of Man (British Isles). The road is characterised by several slopes and elevation variations, thus a three-dimensional [1] model of the track is required in order to perform realistic simulations. The road model used here is the full $3D$ road described in appendix A, which takes into account for all three curvatures κ, ν, τ. In order to perform any lap time simulation, it is necessary to know the $3D$ road geometry, i.e the road middle lane $\mathcal{C}(s)$, the road width $r_w(s)$ and the rotation angles ψ, σ, β or, alternatively, the road curvatures κ, ν, τ. The road geometry has been reconstructed through the procedure described in appendix B. Road borders have been gathered from *Google* satellite and the elevation data has been then added using an on-line tool (`www.gpsvisualizer.com`). Since these tools do not provide such an accuracy to distinguish the different elevation between the right and left road borders, the two borders have been assumed to have the same elevation, i.e. the road banking angle β has been considered to be zero. The raw road data obtained with these tools is shown in figure 5.1; the green markers indicate the value of the estimated curvilinear abscissa s every 10 kilometres.

The road resulting from the reconstruction process is shown in figure 5.2 focusing on a small part of the course comprised between $40.95km < s < 41.4km$. It can be noted that the resulting road borders (blue lines) well match the input points (red crosses). At the same time, the road elevation, the $x - y$ curvature κ and the slope show smooth trends. In order to verify the correctness of the reconstructed road data, it has been compared with the experimental trajectory acquired through a GPS sensor during the practice session at the 2015TT Zero event. Figure 5.3 focuses on three different turns: it can be noticed

[1] The term “three-dimensional” road is here used improperly since the road actually is a two-dimensional manifold. Three-dimensional means that the road is not flat and it extends on a three-dimensional space, i.e. the road presents elevation variations.

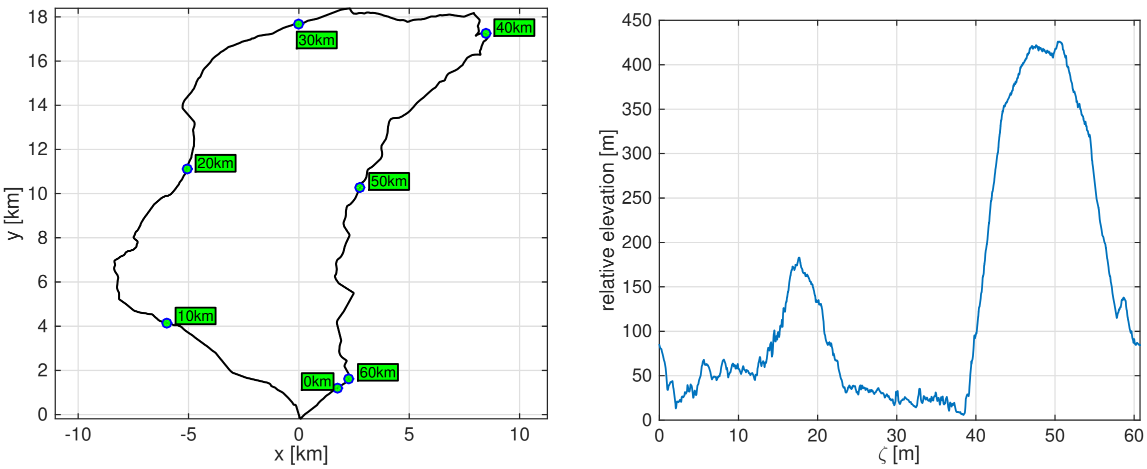


Figure 5.1: Illustration of the Snaefell Mountain Course road data. The $x - y$ coordinates (left image) have been acquired from *Google* satellite images, while the elevation profile was derived via an on-line tool `www.gpsvisualizer.com`. In the left picture the $x - y$ view is represented. The green markers indicate the value of the estimated curvilinear abscissa s every 10 kilometres. The right one shows the relative elevation of the TT course. The elevation peak in correspondence of the Snaefell Mountain is evident.

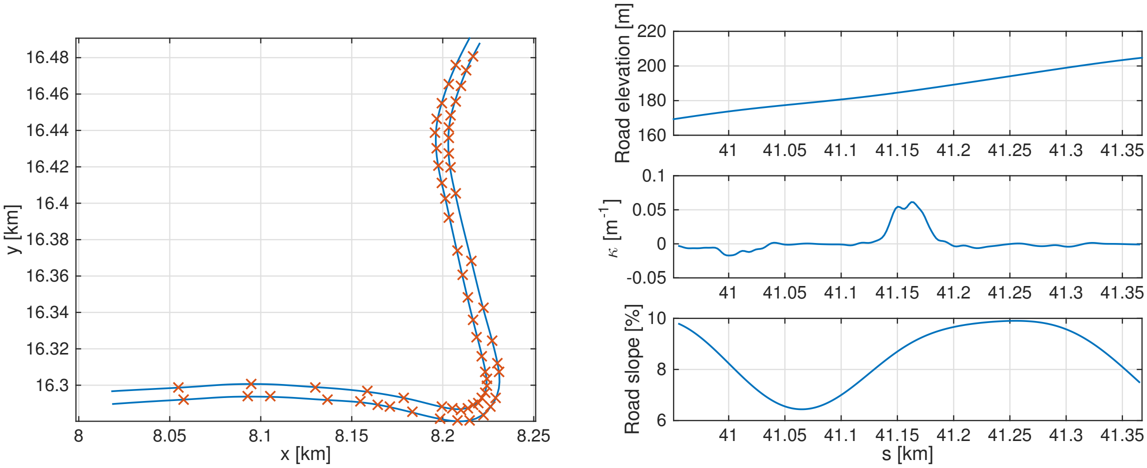


Figure 5.2: The road geometry obtained through solving the optimal control problem is shown here for a small part of the course in the range $40.95km < s < 41.4km$. The left image shows the road input data (red crosses) and borders resulting from the optimal control problem (blue lines) for the part of road comprised between $40.95km < s < 41.4km$. Road elevation, $x - y$ curvature κ and slope percentage are shown in the figure one the right.

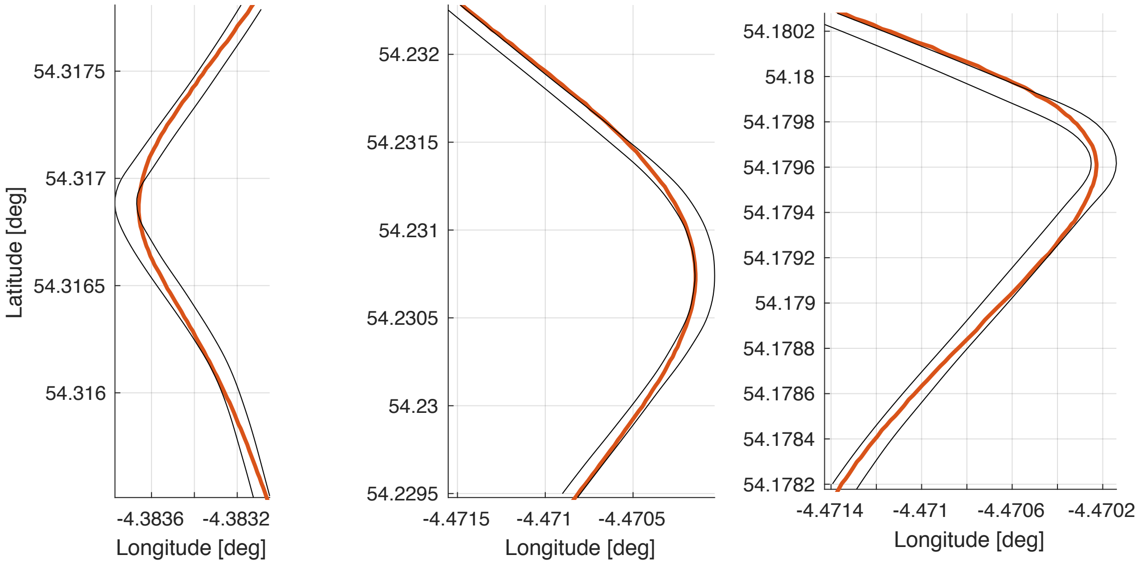


Figure 5.3: Focus on three different turns of the TT road: the black lines are the reconstructed road borders, the orange line is the motorbike trajectory acquired by the GPS sensor. The experimental data confirms the correctness of the road reconstruction procedure.

that, considering the uncertainty in the GPS signal, the road borders have been accurately reconstructed. The experimental data thus confirm the validity of the road reconstruction process and justify the use of the so reconstructed road for minimum time simulations on the TT circuit.

5.3 Motorcycle model

Since the TT Mountain Course is approximately $60km$ long, a simple model is required to execute optimal control simulations over such a long distance in a reasonable amount of time. The use of a motorcycle model with several degrees of freedom over such a long road can make the resulting optimal control problem difficult to solve, leading to long (e.g. hours) computational times or even to a numerically non solvable problem [20, 11]. Thus accurate dynamics modelling is here forgone in order to keep the model simple, yet at the same time a non-trivial power-train model has to be used. The simplest way to describe a vehicle running while staying in contact with the road is a single body that moves on the road plane; however, for a motorcycle the roll motion is very important because it has a remarkable influence on the dynamics. Therefore, the most simple model that captures the essence of the motorcycle dynamics (i.e. g-g diagram and roll motion) is a body that can

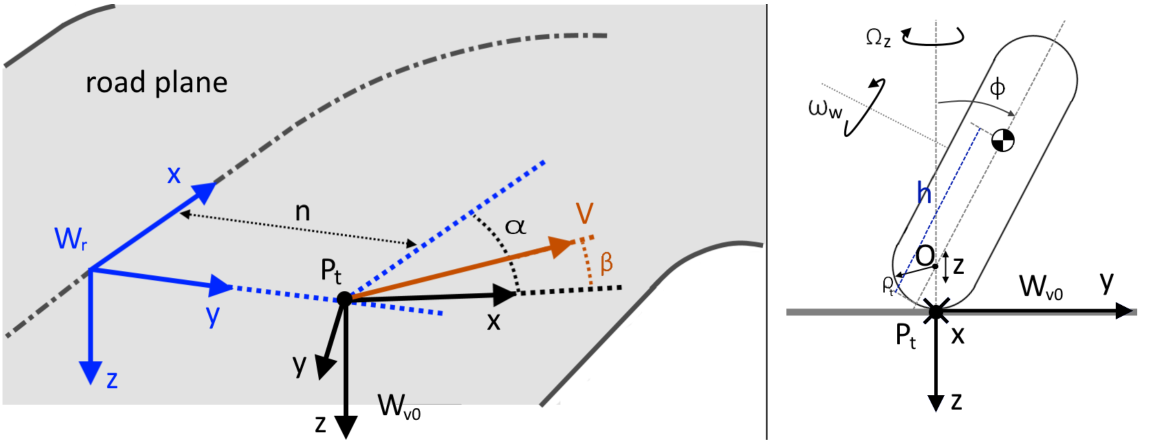


Figure 5.4: Illustration of the transformations from the road frame $\boldsymbol{W}_r$ to the vehicle frame $\boldsymbol{W}_v$. On the left image, the lateral displacement from the centre line n and the relative heading α of the frame $\boldsymbol{W}_{v0}$ are shown. On the right image, the mono-wheel is shown with respect the frame $\boldsymbol{W}_{v0}$. The yaw rate Ω, the roll angle ϕ, the vertical displacement z of the frame $\boldsymbol{W}_v$ and the wheel spin ω_w are highlighted.

move and roll on the road plane, plus a rotating body which accounts for the gyroscopic effects. Such bodies can be thought of as a mono-wheel moving on the road plane, where its mass and inertia tensor are those of the motorcycle.

With reference to the road model described in appendix A, the motorcycle chassis position and orientation w.r.t the road frame $\boldsymbol{W}_r$ are described by means of four variables: the lateral displacement from the centre line n, the relative heading α (see figure 5.4), the vertical displacement z, and the roll angle ϕ. In particular, the relative frame $\boldsymbol{W}_{rel}$ is given by the following sequence of transformations: a lateral displacement of n, a rotation about the z-axis of α, a vertical translation of $\rho_t - z$ (ρ_t is the rear tyre toroid radius), a roll rotation of ϕ and finally a vertical translation of $h - \rho_t$ to reach the CoM position. The frame $\boldsymbol{W}_{rel}$ is thus equal to:

$$\begin{aligned}\boldsymbol{W}_{rel} &= \big(\boldsymbol{T}(0,n,0)\boldsymbol{R}_z(\alpha)\big)\,\big(\boldsymbol{T}(0,0,\rho_t - z)\boldsymbol{R}_x(\phi)\boldsymbol{T}(0,0,\rho_t - h)\big)\\ &= \begin{bmatrix}\cos\alpha & -\sin\alpha & 0 & 0\\ \sin\alpha & \cos\alpha & 0 & n\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & \cos\phi & -\sin\phi & (h-\rho_t)\sin\phi\\ 0 & \sin\phi & \cos\phi & z-\rho_t-\cos\phi(h-\rho_t)\\ 0 & 0 & 0 & 1\end{bmatrix}\end{aligned} \tag{5.1}$$

Similarly to what has been done for the car models in the previous chapter, it is convenient to define the frame $\boldsymbol{W}_{v0}$ that follows only the lateral displacement and the relative heading of the vehicle:

$$\begin{aligned}\boldsymbol{W}_{v0} &= \boldsymbol{W}_r\boldsymbol{T}(0,n,0)\boldsymbol{R}_z(\alpha)\\ &= \boldsymbol{W}_r\begin{bmatrix}\cos\alpha & -\sin\alpha & 0 & 0\\ \sin\alpha & \cos\alpha & 0 & n\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}\end{aligned}\tag{5.2}$$

It can be noted that the roll rotation of the angle ϕ is about an axis parallel to the x-axis of the frame $\boldsymbol{W}_{v0}$, but passing through the centre of the wheel toroid (the point O in figure 5.4), so the contact point between the wheel and the tarmac is always in the origin of $\boldsymbol{W}_{v0}$. Moreover, the mono-wheel can spin with angular velocity ω_w (positive when the motorcycle moves forward) about the wheel centre, for a total of six degrees of freedom.

The mono-wheel model used here is tracked along the road by five dof: s, n, z, ϕ and α. For the derivation of the vehicle equations of motion (i.e. the equations associated to these five dof), the approach described in appendix A could be adopted, yet here a different approach is used avoiding any algebraic equation [2] . In particular, the speed V, sideslip angle λ and yaw rate Ω_{v0z} (see figure 5.4) of the frame $\boldsymbol{W}_{v0}$ have been used as velocity variables instead of the velocities of the vehicle frame $\boldsymbol{W}_v$. The speed V lies in the road surface and the yaw rate $\Omega_{v0z}O$ (and the vertical displacement z too) is aligned with the z-axis of the frame $\boldsymbol{W}_{v0}$, i.e. it is perpendicular to the road plane. The first order equations describing the time variation of the variables s, n and α can be expressed in terms of the speed variables V, λ, Ω_{v0z}, and of the road curvature in the $x-y$ transversal plane κ through the same equations used for the $2D$ roads (4.31)[3]. Since the sideslip angle λ is generally smaller than 10°, the approximations $\sin\lambda \approx \lambda$ and $\cos\lambda \approx 1$ have been adopted.

Since the equations associated to the remaining variables z, V, λ, ϕ and ω_w are derived within the frame $\boldsymbol{W}_{v0}$, it is useful to give the expression of the angular velocity $\boldsymbol{\Omega}_{v0} = \left[\Omega_{v0x}, \Omega_{v0y}, \Omega_{v0z}\right]$ and of the velocity $\boldsymbol{V}_{v0} = \left[V_{v0x}, V_{v0y}, V_{v0z}\right]$ of the frame $\boldsymbol{W}_{v0}$. All of them can be expressed as a function of the state variables and of road geometry through the following relationship:

$$\boldsymbol{\Omega}_{v0} = \begin{bmatrix}\Omega_{v0x}\\ \Omega_{v0y}\\ \Omega_{v0z}\end{bmatrix} = \begin{bmatrix}\dot{s}(\nu\sin\alpha + \tau\cos\alpha)\\ \dot{s}(\nu\cos\alpha - \tau\sin\alpha)\\ \Omega_{v0z}\end{bmatrix}\tag{5.3a}$$

[2] At the time of this work, the available *Pins* version could not handle any algebraic constraint in the OCP formulation.

[3] Since the frame $\boldsymbol{W}_{v0}$ lies on the road surface, its motion is indeed bi-dimensional and, since its velocities has been used as state variables, the tracking equations for the frame $\boldsymbol{W}_{v0}$ are the ones used for the $2D$ road models.

$$\boldsymbol{V}_{v0} = \begin{bmatrix} V_{v0x} \\ V_{v0y} \\ V_{v0z} \end{bmatrix} = \begin{bmatrix} V \\ -V\lambda \\ \dot{s}\tau n \end{bmatrix} \tag{5.3b}$$

It should be noted that the velocity $\boldsymbol{V}_{v0}$ and the angular velocity $\boldsymbol{\Omega}_{v0}$ of the frame $\boldsymbol{W}_{v0}$ in (5.3) can be fully expressed as function of state variables by substituting $\dot{s}$ in (5.3) with expression (4.31a). The Newton equations along the three axis of the frame $\boldsymbol{W}_{v0}$ that describe the time evolution of V, λ, z, are:

$$\begin{aligned}
m\dot{V}_{v0x} + [\cos\phi(-h+\rho_t) - \rho_t + z]m\dot{\Omega}_{v0y} \\
-2m(h-\rho_t)(\Omega_{v0z}\cos\phi - \Omega_{v0y}\sin\phi)\dot{\phi} \\
-m\sin\phi(h-\rho_t)\dot{\Omega}_{v0z} + 2m\Omega_{v0y}\dot{z} - m\Omega_{v0x}\Omega_{v0z}(h-\rho_t)\cos\phi \\
+m\Omega_{v0x}\Omega_{v0y}(h-\rho_t)\sin\phi \\
+m[(-\rho_t+z)\Omega_{v0x} + V_{v0y}]\Omega_{v0z} + m\Omega_{v0y}V_{v0z} = F_{tx} - F_d - mg\sigma\cos\alpha
\end{aligned} \tag{5.4a}$$

$$\begin{aligned}
m\dot{V}_{v0y} + m\ddot{\phi}\cos(\phi)(h-\rho_t) + m[(h-\rho_t)\cos\phi + \rho_t - z]\dot{\Omega}_{v0x} - m\sin\phi(h-\rho_t)\dot{\phi}^2 \\
-2m\sin\phi\Omega_{v0x}(h-\rho_t)\dot{\phi} - 2m\Omega_{v0x}\dot{z} - m(h-\rho_t)(\Omega_{v0z}^2 + \Omega_{v0x}^2)\sin\phi \\
-m\Omega_{v0y}\Omega_{v0z}(h-\rho_t)\cos\phi - m[(\rho_t - z)\Omega_{v0y} - V_{v0x}]\Omega_{v0z} - m\Omega_{v0x}V_{v0z} \\
= F_{ty} + F_l\sin\phi + mg\sigma\sin\alpha
\end{aligned} \tag{5.4b}$$

$$\begin{aligned}
m\ddot{z} + m\ddot{\phi}\sin(\phi)(h-\rho_t) + m\cos\phi(h-\rho_t)\dot{\phi}^2 \\
+2m\cos\phi\Omega_{v0x}(h-\rho_t)\dot{\phi} \\
+m\sin\phi(h-\rho_t)\dot{\Omega}_{v0x} + m\dot{v}_z + m\cos\phi(h-\rho_t)(\Omega_{v0x}^2 + \Omega_{v0y}^2) \\
+m\Omega_{v0y}\Omega_{v0z}(h-\rho_t)\sin\phi \\
-m(-\rho_t+z)\Omega_{v0x}^2 + m\Omega_{v0x}V_{v0y} - m(-\rho_t+z)\Omega_{v0y}^2 \\
-m\Omega_{v0y}V_{v0x} = mg - N - F_l\cos\phi
\end{aligned} \tag{5.4c}$$

where m is the total mass of the motorcycle plus the rider, h is the reference distance of the centre of mass (CoM) from ground, r_t is the wheel cross section radius, F_{tx}, F_{ty} and N are respectively the wheel longitudinal, lateral and normal force, F_d is the drag force, F_l is the aerodynamic lift force. Since the road slope σ is always smaller than 10°, the approximations $\sin\sigma \approx \sigma$ and $\cos\sigma \approx 1$ have been adopted. Moreover, the Euler equation governing the roll dynamics, calculated with respect to the origin of the frame $\boldsymbol{W}_{v0}$, is the following:

$$\begin{aligned}\{[-(h-\rho_t)(z-\rho_t)\cos\phi+(h-\rho_t)^2]m+I_{xx}\}\ddot{\phi}+I_{xx}\dot{\Omega}_{v0x}\\ +(\Omega_{v0z}\cos\phi-\Omega_{v0y}\sin\phi)I_{gy}\omega_w+\sin\phi(z-\rho_t)F_l\\ +(I_{yy}-I_{zz})[\cos\phi\sin\phi(\Omega_{v0y}^2-\Omega_{v0z}^2)+\Omega_{v0y}\Omega_{v0z}(1-2\cos\phi^2)]+m\tau_\phi=0\end{aligned} \tag{5.5}$$

where I_{xx}, I_{yy}, I_{zz} are the inertia moments of the motorcycle with respect to the Cartesian axes, $I_{gy}=2I_{wy}+I_{ey}\tau_g$ is the gyroscopic moment of inertia, i.e. the sum of the front and rear wheel y-axis inertia moment I_{wy} and the engine y-axis inertia I_{ey}, taking into account the motor to wheel gear ratio τ_g. τ_ϕ is the sum of several terms proportional to the mass of the mono-wheel which appears as consequence that the Euler equation is not written with respect to the centre of gravity. The chosen pole (the origin of the frame $\boldsymbol{W}_{v0}$) has the advantage that tyre forces does not appear in equation (5.5). The expression of τ_ϕ is the following:

$$\begin{aligned}\tau_\phi=&(h-\rho_t)\left[\sin\phi(\ddot{z}-\Omega_{v0y}V_{v0x}+\Omega_{v0x}V_{v0y}+\dot{V}_{v0z}-g)\right.\\ &-\cos\phi(\dot{V}_{v0y}-2\Omega_{v0x}\dot{z}+\Omega_{v0x}V_{v0z}-\Omega_{v0z}V_{v0x}+g\sigma\sin\alpha)]\\ &+(h-\rho_t)(\rho_t-z)[2\cos\phi\dot{\Omega}_{v0x}-2\cos\phi\Omega_{v0z}\Omega_{v0y}+\sin\phi(\Omega_{v0y}^2-\Omega_{v0z}^2-\dot{\phi}^2-2\Omega_{v0x}\dot{\phi})]\\ &+(h-\rho_t)^2[\dot{\Omega}_{v0x}+\sin\phi\cos\phi(\Omega_{v0y}^2-\Omega_{v0z}^2)-2\cos^2\phi\Omega_{v0x}\Omega_{v0y}]\\ &+(z^2-2\rho_t z)\dot{\Omega}_{v0x}+(h-z)(h-2\rho_t+z)\Omega_{v0y}\Omega_{v0z}\\ &+(\rho_t-z)(\dot{V}_{v0y}-V_{v0z}\Omega_{v0x}+V_{v0x}\Omega_{v0z}-2\dot{z}\Omega_{v0x}-g\sigma\sin\alpha)\end{aligned} \tag{5.6}$$

Another equation is required to describe the spin motion of the mono-wheel, which is the Euler equation of the mono-wheel about its y-axis:

$$\begin{aligned}\dot{\omega}_w I_{wy}=&\tau_d-\tau_R-F_{tx}[r_t-\rho_t(1-\cos\phi)-z\cos\phi]\\ &+I_{wy}[\dot{\Omega}_{v0y}\cos\phi+\sin\phi\dot{\Omega}_{v0z}+(\Omega_{v0z}\cos\phi-\Omega_{v0y}\sin\phi)\dot{\phi}]\end{aligned} \tag{5.7}$$

where τ_d is the driving torque applied to the rear wheel, τ_R is the rolling resistance torque and I_{wy} is the inertia moment of the wheel only. The six variables used in equations (5.4), (5.5), (5.7) to capture the six degrees of freedom of the mono-wheel are: $V, \lambda, z, \phi, \Omega_{v0z}, \omega_w$. No equation is used to describe the time evolution of the yaw rate Ω_{v0z} because it is a control input.

The previous equations depend on the external forces and torques, and on the internal driving torque τ_d acting on the motorcycle. The driving torque τ_d is actually the sum of the positive traction motor torque and the negative braking torque:

$$\tau_d = \tau_t + \tau_b \implies \tau_t = f^+(\tau_d), \quad \tau_b = f^-(\tau_d) \tag{5.8}$$

where $f^+(x)$ and $f^-(x)$ are two functions that return the positive and negative part of the argument respectively; the simultaneous usage of the brake and the gas is not allowed in this model. The external forces are the tyre longitudinal force F_{tx}, the tyre lateral force F_{ty}, and the aerodynamic drag F_d and lift F_l; the only external torque is the rolling resistance τ_R. The tyre longitudinal F_{tx} and lateral F_{ty} forces, together with the rolling resistance τ_R and the tyre load N are given by a linear tyre model, limited by an adherence ellipse:

$$\begin{aligned} F_{tx} &= N K_{t\kappa} \kappa_t \\ F_{ty} &= N K_{t\lambda} \lambda_t + K_{t\phi} \phi) \\ \tau_R &= (N \cos\phi + F_{ty} \sin\phi) K_{\tau_R} \\ N &= f^+(K_{tr} z + K_{td} \dot{z}) \end{aligned} \tag{5.9}$$

where $K_{t\kappa}$, $K_{t\lambda}$ and $K_{t\phi}$ are the tyre longitudinal, lateral and roll stiffness, K_{tr} and K_{td} are the tyre radial stiffness and damping stiffness, and finally the function f^+ ensures the tyre load to be non negative. The tyre forces F_{tx} and F_{ty} are forced to lie within the tyre adherence ellipse:

$$\left(\frac{F_{tx}}{N\mu_x}\right)^2 + \left(\frac{F_{ty}}{N\mu_y}\right)^2 \leq 1 \tag{5.10}$$

where μ_x and μ_y are the the maximum longitudinal and lateral tyre adherence respectively. The longitudinal tyre slip, κ_t, is given by:

$$\kappa_t = \frac{\omega_w[r_t - \rho_t + \cos\phi(\rho_t - z)] - V}{V} \tag{5.11}$$

where r is the rear wheel radius. Finally, the drag and lift aerodynamic forces are proportional to the square of the motorcycle speed:

$$F_d = \frac{1}{2}\rho_a c_{dA} V^2 \qquad F_l \quad = \frac{1}{2}\rho_a c_{lA} V^2 \tag{5.12}$$

where ρ_a is the air density, C_d and C_l are the drag and lift coefficients. The numeric values of the motorcycle data used to feed the model are listed in table 5.1.

Table 5.1: Electric motorbike dataset.

symbol	value	units	description
g	9.81	m/s^2	gravitational acceleration
ρ	1.2	kg/m^3	air density
h	0.6	m	centre of gravity (CoG) height
r_t	0.3	m	wheel radius
ρ_t	0.1	m	wheel toroid radius
m	290	kg	motorbike mass including rider and 4 battery packs
I_{xx}	19	kgm^2	motorbike roll moment of inertia
I_{gy}	2.1	kgm^2	gyroscopic moment of inertia
I_{wy}	0.7	kgm^2	wheel spin moment of inertia
$K_{t\lambda}$	12	-	tyre sideslip stiffness
$K_{t\phi}$	1	-	tyre roll stiffness
K_{tr}	2×10^5	N/m	tyre radial stiffness
K_{td}	1×10^3	Ns/m	tyre radial damping
μ_x	0.8	$-$	tyre longitudinal adherence
μ_y	0.6	$-$	tyre lateral adherence
K_{τ_R}	0.015	m	tyre rolling resistance
c_{dA}	0.41	m^2	drag coefficient
c_{lA}	0.03	m^2	lift coefficient

5.4 Powertrain model

For the inaugural TTXGP grand prix in 2009, a wide variety of first designs were brought to the grid. The BX electric motorcycle chassis from the Brunel Racing team carried a simple power train, consisting of a single brushed DC motor, matching controller and a Lithium Ion battery pack. From 2010 and the first TT Zero Challenge onward, the BX power train was upgraded to a dual brushed DC motor configuration, matching controller and various sizes of a Lithium Polymer battery pack. This configuration became a favourite among entries for TTXGP national championships and Isle of Man TT Zero events, and best represents first generation designs. The BX dual motor power train with Li-Po pack from Brunel Racing was developed year on year until the 2015 TT Zero event, and it is this latest version BX-15 that has been modelled here.

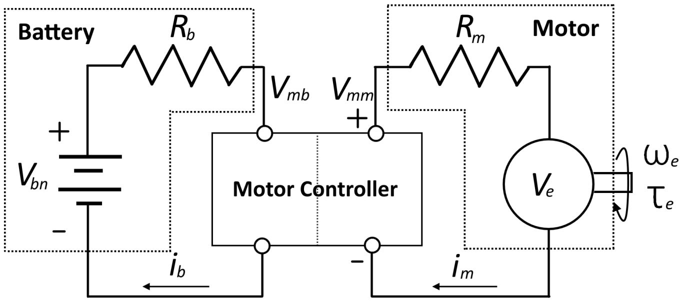


Figure 5.5: Equivalent electric circuits of battery and electric motor.

The BX-15 power train is composed of two customized Agni 95R brushed DC and axial flux motors, supplied by a bespoke Kokam Li-Po battery pack in “4P24S” (4 parallel of 24 series) cell arrangement and governed by an off-the-shelf 1200A Kelly KDHE controller. The two motors have been balanced so that the total power delivery is equally split over each one. The electrical equivalent model used to describe the behaviour of such an electric power train is given in figure 5.5 and is composed of two main circuits, a battery circuit and a motor circuit, connected via a motor controller that drives the motor voltage/current input so as to deliver the requested torque at the motor output shaft.

The two motors in the power train have been modelled via a single motor circuit: indeed, two motors working in parallel can be seen as equivalent to a single motor of the same characteristics but half the internal resistance. The battery pack is modelled via a battery circuit, composed of an ideal voltage supply V_{bn} in series with a resistor R_b that accounts for the battery pack's internal resistance [43]. Output voltage Vbn decreases linearly with battery discharge status, from V_{bn1} when the battery pack is fully charged to V_{bn0} when it is fully discharged. Therefore, the closed loop battery voltage V_{mb} may be calculated as follows:

$$V_{mb} = V_{bn} - i_b R_b \tag{5.13a}$$

$$V_{bn} = V_{bn0} + (V_{bn1} - V_{bn0})\frac{e}{e_i} \tag{5.13b}$$

where i_b is the current flowing in the batteries, e and e_i are respectively the actual and initial battery charge status. The motor controller is a DC to DC converter from the battery circuit to the motor circuit, and for simplicity the conversion efficiency η_{mc} is assumed to be constant, yielding to the following power balance equation:

$$V_{mm}i_m = \eta_{mc}V_{mb}i_b \tag{5.14}$$

where V_{mm} is the motor controller voltage output to the motor circuit which is related to the current i_m circulating in the motor and the the self-induced or back electromagnetic force (e.m.f.) V_e. Indeed, in steady condition the voltage balance is:

$$V_{mm} = i_m R_m(T) + V_e \tag{5.15}$$

where the R_m represents the armature winding resistance, which depends on motor temperature T. Finally, the conversion of the electric power into mechanical power is governed by the well known proportional equations [43, 45]:

$$V_e = \kappa_V(T)\omega_m \qquad \tau_e = \kappa_\tau(T)f^+(i_m - i_{m0}) \tag{5.16}$$

where ω_m is the motor rotational speed, i_{m0} is the engine current offset, τ_e the motor torque, $\kappa_\tau(T)$ the torque constant and $\kappa_V(T)$ the back e.m.f constant, with $\kappa_\tau(T) < \kappa_V(T)$ because of eddy current and other conversion losses.

In the equations (5.15) and (5.16) three motor characteristics depend on the motor temperature: the armature winding resistance R_m and the just mentioned proportional constants κ_τ, κ_V. Specifically, R_m increases with motor temperature while $\kappa_\tau(T)$, $\kappa_V(T)$ diminish because the hotter the motor, the lower the magnetic flux of the permanent magnet is. A linear relationship has been used to capture the variation with temperature of these three quantities:

$$\begin{aligned} R_m(T) &= R_{m0} + \sigma_R(T - T_0) \\ \kappa_V(T) &= \kappa_{V0} - \sigma_m(T - T_0) \\ \kappa_\tau(T) &= \kappa_{\tau 0} - \sigma_m(T - T_0) \end{aligned} \tag{5.17}$$

where T_0 is the corresponing temperature for the reference values R_{m0}, κ_{V0} and $\kappa_{\tau 0}$.

Since motor characteristics vary with motor temperature, another equation is required to describe the motor temperature variation over time. The motor heat balance allows to calculate the motor temperature rate on the basis of incoming Q_{in} and outgoing Q_{out} heat flux:

$$C\dot{T} = Q_{in} - Q_{out} \tag{5.18}$$

where

$$Q_{in} = V_{mm} i_m - \omega_m \tau_e \qquad Q_{out} \quad = q(T - T_{ext}) \tag{5.19}$$

where C is the motor thermal capacity, T_{ext} is the environmental temperature and q is the conduction coefficient, which is asusmed to vary linearly from q_0 when the motorcycle is stopped, to q_1 at a speed $V_1 = 150 km/h$:

$$q = q_0 + \frac{q_1 - q_0}{V_1} V \tag{5.20}$$

It is worth pointing out that no dynamic effects have been included in the model of the electric power train: indeed, electromagnetic transients are much faster than mechanical ones and the former may be neglected for our purposes. In other words, for a step variation of the controlled voltage V_{mm} (or current, i_m) it is reasonable to assume that the current i_m (or voltage, V_{mm}) as well as the motor torque τ_e vary instantaneously, while the variation of the motor speed ω_m and e.m.f. V_e are governed by vehicle inertia through equation (5.7).

In conclusion using equations (5.15) and (5.16), the controller voltage V_{mm} can be expressed as function of the motor torque τ_e and motor speed ω_e

$$V_{mm} = V_e + i_m R_m = \omega_e \kappa_V + \frac{\tau_e}{\kappa_\tau} R_m \tag{5.21}$$

Moreover, taking into account the powertrain efficiency η_c and the gear ratio τ_g between the motor speed ω_e and wheel speed ω ($\omega_e = \omega_w \tau_g$), it is possible to express τ_e as a function of the wheel traction torque τ_t:

$$\tau_e = \frac{\tau_t}{\eta_c \tau_g} \tag{5.22}$$

Using equations (5.13) and (5.14) it is also possible to obtain an expression for the battery current i_b and motor controller voltage at the battery side V_{mb} as function of the motor torque τ_e and speed ω_e

$$\begin{aligned} i_b &= \frac{V_{bn}}{2R_b} - \frac{\sqrt{\eta_m (V_{bn}^2 \eta_m - 4(\frac{\tau_e}{\kappa_\tau})^2 R_b R_m - 4\frac{\tau_e}{\kappa_\tau} R_b \omega_e \kappa_V)}}{2R_b \eta_m} \\ V_{mb} &= V_{bn} - i_b R_b \end{aligned} \tag{5.23}$$

The expression of i_b is useful because it allows to evaluate the discharge rate of the battery charge:

$$\dot{e} = -V_{bn} i_b \tag{5.24}$$

Also, efficiency values both of the motor only and of the power train i.e. motor, battery pack and motor controller, are shown in figure 5.6 as a function of motor speed and motor torque. It can be noticed that the brushed DC motors used in BX-15 are characterised by an efficiency up to ≈ 85%, and that further losses in the complete power train drop the efficiency value by an additional ≈ 5%.

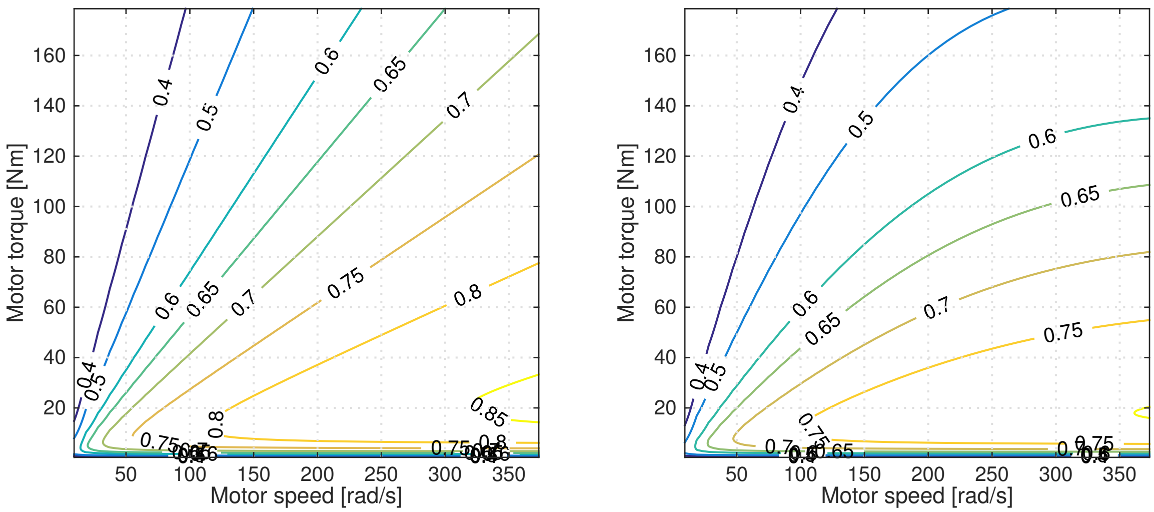


Figure 5.6: Contour plot of efficiency values as a function of the motor speed and motor torque. The left plot refers to the motor efficiency only, while the right plot shows overall power train efficiency, including motor, battery pack and motor controller. Data shown was derived for a motor temperature of $25°C$.

Finally, it should be noted that the only variables that the electric motor model adds to the state space model are the motor temperature T and the battery charge e. All other quantities are algebraic expressions of other variables and controls.

5.5 State space and OCP formulation

The vehicle dynamics and electric power train equations described in the previous sections can be put together to form a first order ordinary differential equation system that completely characterises the model. In order to do this, two auxiliary variables have to be introduced because both roll angle ϕ and the vertical displacement z appear respectively in equation (5.5) and (5.4) with the second order derivative:

$$\dot{\phi} = \phi_{dot} \qquad \dot{z} = z_{dot} \tag{5.25}$$

At this point, equations (4.31), (5.4), (5.5), (5.7), (5.18), (5.24), (5.25) completely describe the mono-wheel dynamics as a system of 12 first order differential equations with as many state variables $\mathbf{x}_s$ and 2 inputs $\mathbf{u}_s$:

$$\begin{aligned} \boldsymbol{x}_s &= \Big[s; \quad n; \quad \alpha; \quad V; \quad \lambda; \quad z; \quad \phi; \quad \omega_w; \quad T; \quad e; \quad \phi_{dot}; \quad z_{dot} \Big] \\ \boldsymbol{u}_s &= \Big[\Omega_{v0z}; \quad \tau_d \Big] \end{aligned} \tag{5.26}$$

Recalling that the motor torque τ_e can be easily expressed as function of the control τ_d (5.22), all other electric variables as V_{mm}, i_m, i_b, V_b can be expressed as function of τ_d and the state variable ω_w using equations (5.16) (5.15) (5.23).

The minimum time problem is here formulated as in section 4.2.5, 4.3.2 and 4.4.2, thus it is only briefly described. The transformation from time to space domain described in appendix A is adopted, thus the variable s is made the independent variable and few constraints are used to make the simulation withstand the real racing conditions. In particular such constraints ensure that:

(a) the motorcycle never exceeds the road borders, considering that the track has width $r_w(s)$;

(b) the roll rate is lower than the maximum one real driver can handle $\phi_{dot}^{(max)}$;

(c) the yaw rate is less than the one real motorcycle can achieve $\Omega^{(max)}$ (this constraint has to be included because the yaw rate is a control);

(d) the tyre forces never exceed the adherence ellipse, as already discussed in section 5.3 (the maximum longitudinal and lateral tyre adherence are μ_x, μ_y respectively);

(e) the motor current remains lower than the maximum motor current $i_m^{(max)}$;

(f) the power absorbed by the motor controller from the battery pack is lower than the maximum battery pack power $P_b^{(max)}$;

(g) the battery charge e does not become negative.

Such conditions can be expressed by the following relationships:

$$r_w/2 \leq n \leq r_w/2 \tag{5.27a}$$

$$-\phi_{dot}^{(max)} \leq \phi_{dot} \leq \phi_{dot}^{(max)} \tag{5.27b}$$

$$-\Omega^{(max)} \leq \Omega_{v0z} \leq \Omega^{(max)} \tag{5.27c}$$

$$\left(\frac{F_{tx}}{N \mu_x} \right)^2 + \left(\frac{F_{ty}}{N \mu_y} \right)^2 \leq 1 \tag{5.27d}$$

$$i_m \leq i_m^{(max)} \tag{5.27e}$$

$$V_{bn} i_b \leq P_b^{(max)} \tag{5.27f}$$

$$e \geq 0 \tag{5.27g}$$

The *Pins* software described in chapter 3.2.1 has been used as numerical solver.

5.6 Simulation results and model validation

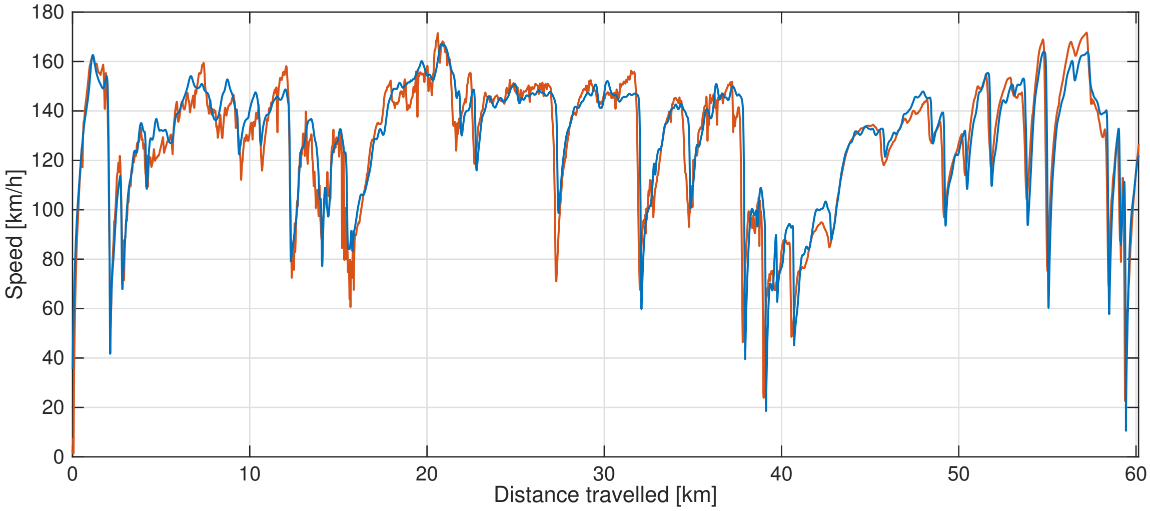


Figure 5.7: Experimental (red) and simulated (blue) speed profile versus distance travelled.

The minimum time optimal control simulation has been carried out on the entire Snaefell Mountain Course obtaining a theoretical lap time of 29 minutes and 31 seconds, which is in good agreement with the physical lap of 29 minutes and 47 seconds, registered during a practice session of the 2015TT Zero event. The simulation took less than three minutes to compute on a desktop PC equipped with an *Intel Core i7* processor, using a discretisation mesh with one point every meter (leading to a total of $\approx 60 \times 10^3$ mesh points); this suggest that the used optimal control solver, together with the motorcycle model presented in this work, is very efficient. The simulated motorcycle speed along the track is shown in figure 5.7 together with the experimental one; the figure highlights a good qualitative agreement between the experimental data and the simulation. It can be observed that the simulation is slightly faster than the actual rider between the 40^{th} and 50^{th} kilometre. This is the section of the course with the most significant positive slope (ascent of the Snaefell Mountain) and, as the bottom graph of figure 5.8 suggests, demands more power in simulation. As the real rider had some freedom not to apply maximum power over the entire course so as to prevent excessive motor load and preserve battery pack energy (even if battery pack capacity was

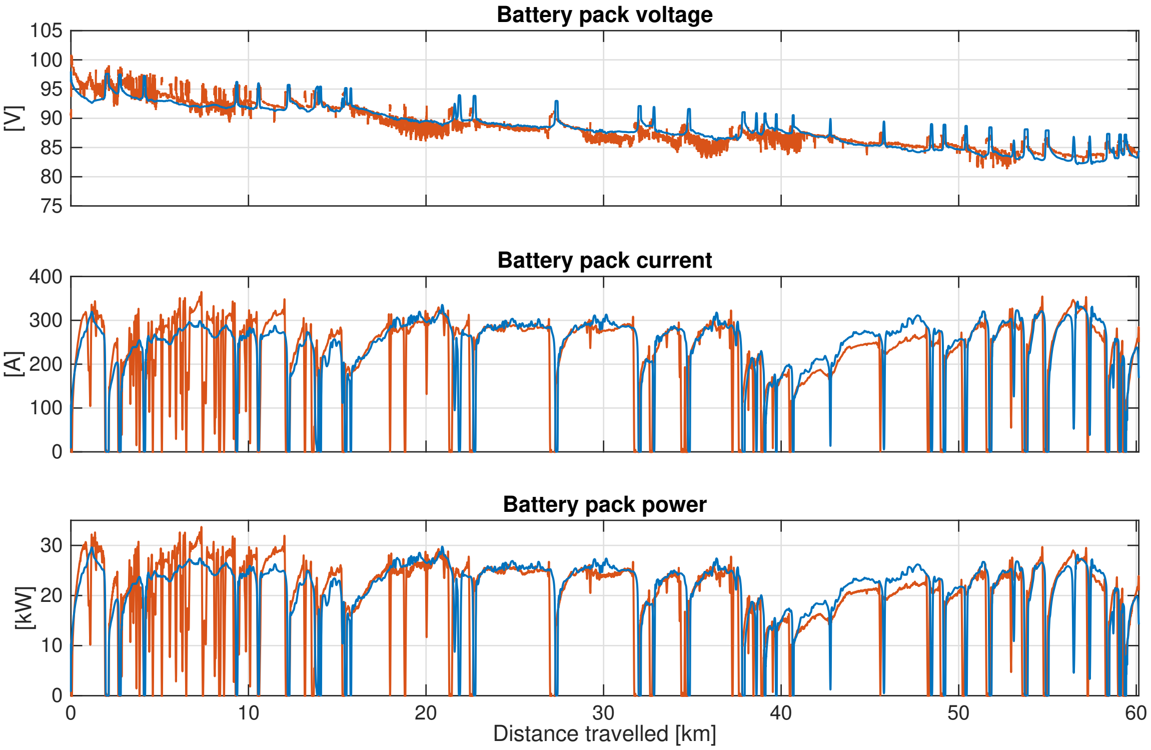


Figure 5.8: Battery pack voltage (top graph), current (centre graph) and power (bottom graph) vs. distance travelled. Red lines refer to experimental data, blue lines to simulation.

sized with contingency to avoid full battery discharge), this may be the reason why the rider navigated the final climbing section of the course somewhat slower than he could, as optimal control simulation indicates.

Figure 5.8 reports on the energy usage along the course: battery pack voltage, current and power delivery in simulation are shown together with the experimental data, and demonstrate very good agreement for all three parameters. The simulated battery pack voltage (top graph) agrees well with the experimental data, capturing not only the expected decreasing trend of the battery discharge process but also the location and duration of voltage recovery effects when load is being removed. These voltage effects align with the current and power drops observed in the centre and bottom graphs respectively, as the rider releases the throttle. The simulated battery pack current draw and power drain further match well with current and power as measured on track, as well as with the speed graphs from figure 5.7, only showing slightly greater values for the first 15 kilometres of the course and slightly lower values between the 40^{th} and 50^{th} kilometre, indicating that the rider was pushing beyond optimal energy drain at the start and started to preserve energy near the end. It can be expected that there are additional smaller effects impacting

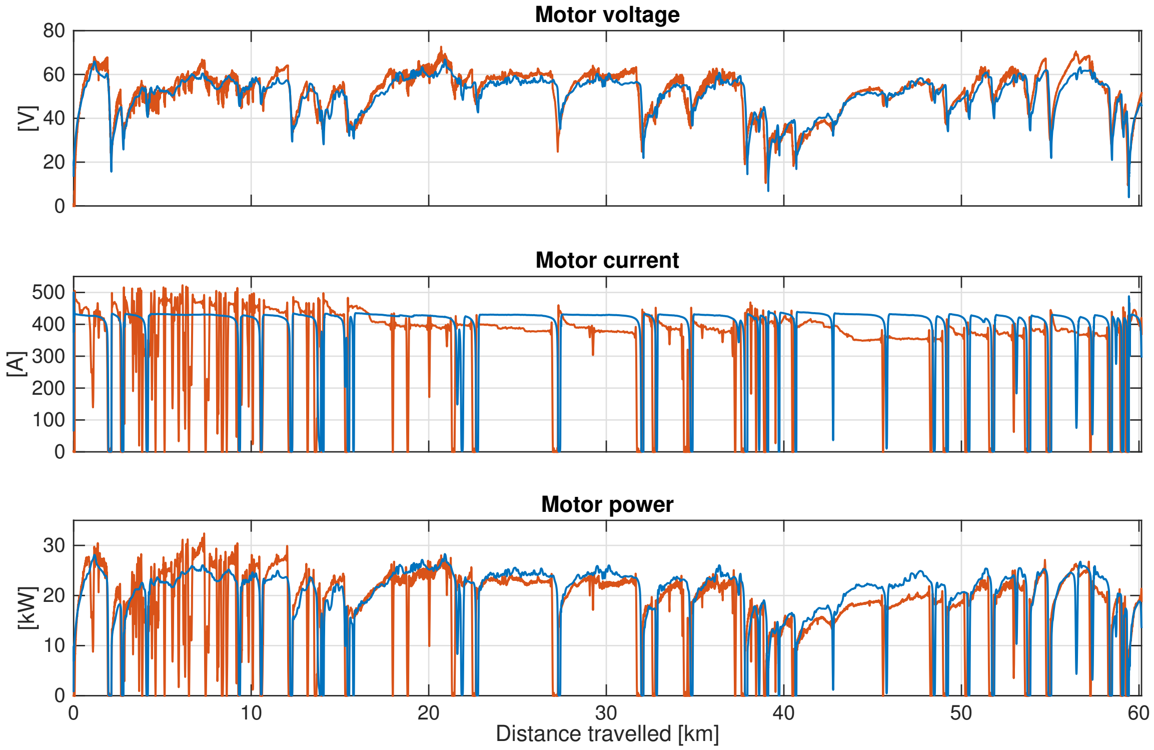


Figure 5.9: Motor voltage (top graph), current (centre graph) and power consumption (bottom graph) vs. distance travelled. Red lines refer to experimental data, blue lines to simulation.

on power drain and efficiency near the start and towards the final heavy load stages of the race, such as the cold start and high final temperatures, and the fact that battery charge status is always slightly above nominal in the beginning and close to non-linear drop-off for individual battery cells near the final stages. Even though the experimental data could not provide reliable specific temperature nor individual battery cell information to quantify the impact of such factors, the fact that the deviations observed between simulation and experiment in figure 5.8 occur only near the start and end of the race distance, exactly when both the rider would be adjusting his riding style and the aforementioned temperature and individual battery characteristics are most dominant, underlines that the modelling approach is sound.

Figure 5.9 shows the motor related variables. The simulated motor current (centre graph) however presents some differences with the experimental data. In the first part of the TT course, the experimental motor current is, on average, higher than the simulated current, while in the second part the opposite can be observed. The same behaviour can be noticed for the power absorbed by the motor, which is a consequence of the current

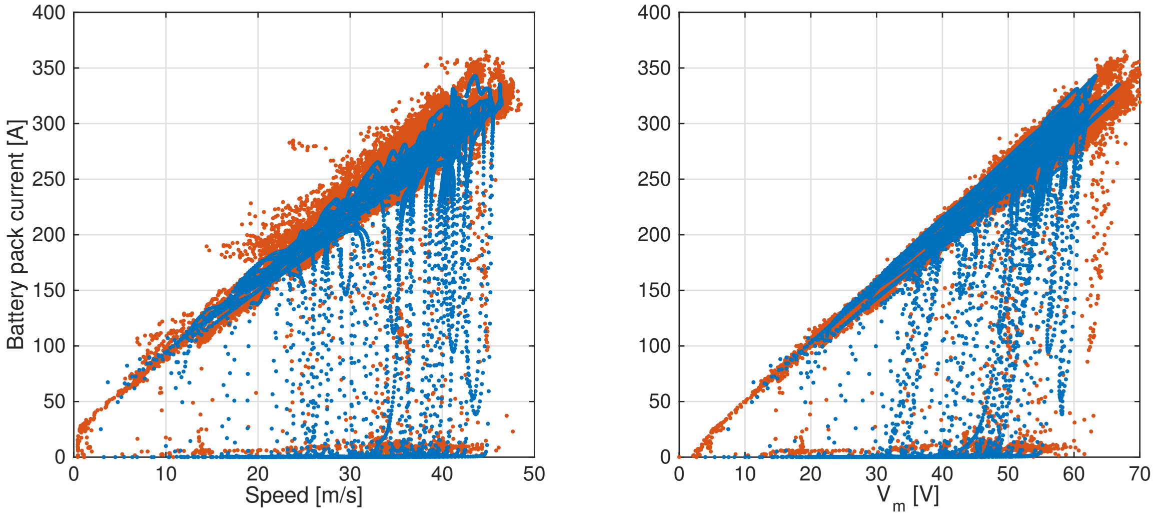


Figure 5.10: Battery pack current draw vs. motorcycle speed (left) and motor driving voltage (right). The simulated data (blue dots) shows the same linear trend as the experimental data (red dots).

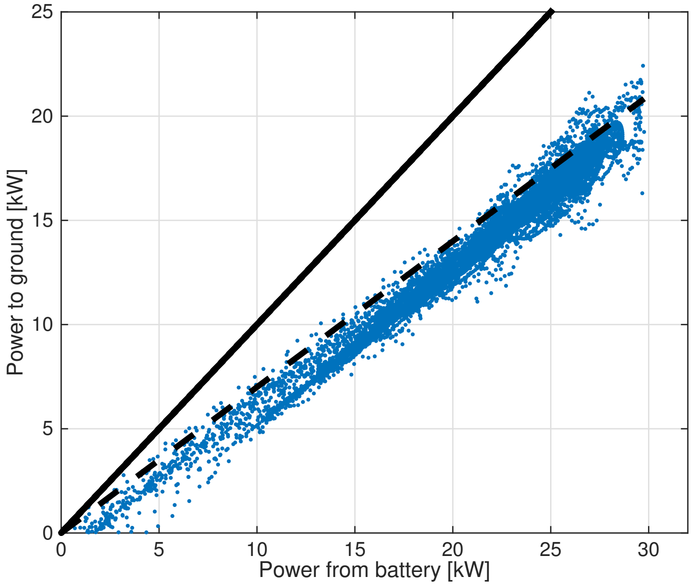


Figure 5.11: Power delivered to the ground versus power drained from battery (simulated data). The black solid and dashed lines correspond respectively to an overall efficiency of the 100% and 70%: the simulated motorcycle highlights an efficiency close to the 70%.

deviations as power is calculated via the product of motor voltage and motor current. This discrepancy can be due to the uncertainty in the measurement of motor current: indeed, the motorcycle was equipped with two motors working in parallel, yet only the current drawn by the left motor has been captured due to a failure in the right motor sensor. Total motor current had to be estimated by multiplying the left current by two, as the dual motor configuration had been balanced at the start to have the two motors working equally. However, not only is it known that each brushed DC Agni 95R motor is subject to a degree of variation from the nominal specification due to the non-automated production process applied (affecting armature internal resistance for example), balancing of multiple motor configurations also has to performed track side, especially after servicing that must occur after each track outing. The balancing process consists of manual tuning (i.e. not via motor controller programming) of the brush holder setup of each motor in the configuration, and is therefore prone to slight deviations around the ideal brush holder positions, which carries no impact for normal road and short circuit race applications but could affect individual motor temperature, power and efficiency, and therefore motor balancing within a multiple configuration, when under sustained heavy loading over a long distance such as around the TT Snaefell Mountain Course. And indeed, the two motors used here did turn out to have an asymmetric loading along the course, as found from temperature observations. Even though the balanced motors were mounted to ensure the same exposure to ambient air cooling, different working temperatures of $\approx 30°$ for the left and $\approx 50°$ for the right motor were observed. Such discrepancy suggests that an asymmetric motor loading existed along the course, which was not incorporated in the simulation where an average working temperature for a balanced motor model was taken at $\approx 40°$.

Further evidence that the mathematical models of motorcycle and electric power train are reliable can be taken from figure 5.10. The left graph shows a linear trend between simulated battery pack current draw and motorcycle speed, which corresponds well with experimental observations. The same holds true for the battery pack current and the motor driving voltage on the right graph, as would be expected since motor speed and motor voltage are proportionally related for brushed DC motors.

Finally, figure 5.11 illustrates from simulation data the calculated overall efficiency of the complete energy conversion process: the mechanical power to ground is plotted as a function of the energy drained from the battery pack. The black lines correspond to overall efficiencies of 100% (solid) and 70% (dashed). The simulated motorcycle (blue dots) demonstrates an average overall efficiency close to or slightly below the 70% marker line.

5.7 Motorbike optimisation

The previous section has demonstrated the ability of the mathematical model to faithfully reproduce the entire run on the TT course, suggesting that simulation can now also be used to optimise the motorcycle design for the specific course. Lap time simulation can be effective in quickly highlighting performance variations, for example during selection of motorcycle configuration or race setup parameters for which optimal values are not easily identified other than by experience or extensive historical data. In this section the value of optimisation via lap time simulation for the design and setup alterations of three important electric motorcycle parameters will be demonstrated: the wheel-to-motor gear ratio, the battery pack capacity, and the number of motors within the power train configuration will be optimised.

The wheel-to-motor gear ratio mainly influences the motorcycle performance because it modifies both wheel torque and top speed. Since maximum motor torque and motor speed result from motor design and build and thus are fixed, a shorter wheel-to-motor gear ratio allows to reach higher wheel torque, and therefore faster motorcycle accelerations, yet at the same time it reduces maximum motorcycle speed. Moreover, figure 5.6 showed that, for a given motor power, the overall power train efficiency of motor, controller and battery pack increases as motor torque decreases, except for very low motor torques. On the basis of these considerations it is not immediately clear which gear ratio should be best used for the TT course. Different runs on the TT course have been simulated varying the wheel-to-motor gear ratio in the range 2.5 to 4.3, with step changes of 0.1; all the other characteristics of the motorcycle were kept unchanged. Results are shown in figure 5.12 and compared with reference values from the data and setup of the (non-optimized) BX-15 chassis, as detailed in section 5.4. It can be noticed that the optimal lap time is approximately 1 minute and 45 seconds ($\approx 6\%$) less than the reference time, and it is obtained for a gear ratio of 3.4, which is significantly higher than the reference gear ratio 2.8. A lap time reduction of this magnitude corresponds to an increase in average motorcycle lap speed from $122km/h$ ($75.8mph$) to $130km/h$ ($80.8mph$). It can be observed that the optimal gear ratio provides neither maximum top speed[4], nor maximum $0 - 120km/h$ acceleration. In fact, maximum top speed is achieved with a gear ratio equal to 3.1, while maximum acceleration is obtained with the shortest gear ratio considered (4.3). Figure 5.12 further shows that the gear ratio influences significantly the amount of energy used over the entire course (fourth graph from the top): the maximum energy usage ($41MJ$ or $11.4kWh$) is obtained with a gear ratio equal to 3.3, which is close to the optimal one, while the reference data shows approximately 10% less energy usage ($37MJ$ or $10.3kWh$) with gear ratio 2.8. None of the gear ratios considered here would fully drain the battery pack; indeed the BX-15 battery pack size was intentionally over-dimensioned on safety grounds to prevent full discharge. Moreover, the bottom graph in figure 5.12 shows that the overall efficiency, calculated as the total energy delivered to the wheel divided by the battery pack energy used, increases continuously with

[4] It is the maximum speed achieved on the course, not the motorcycle top speed.

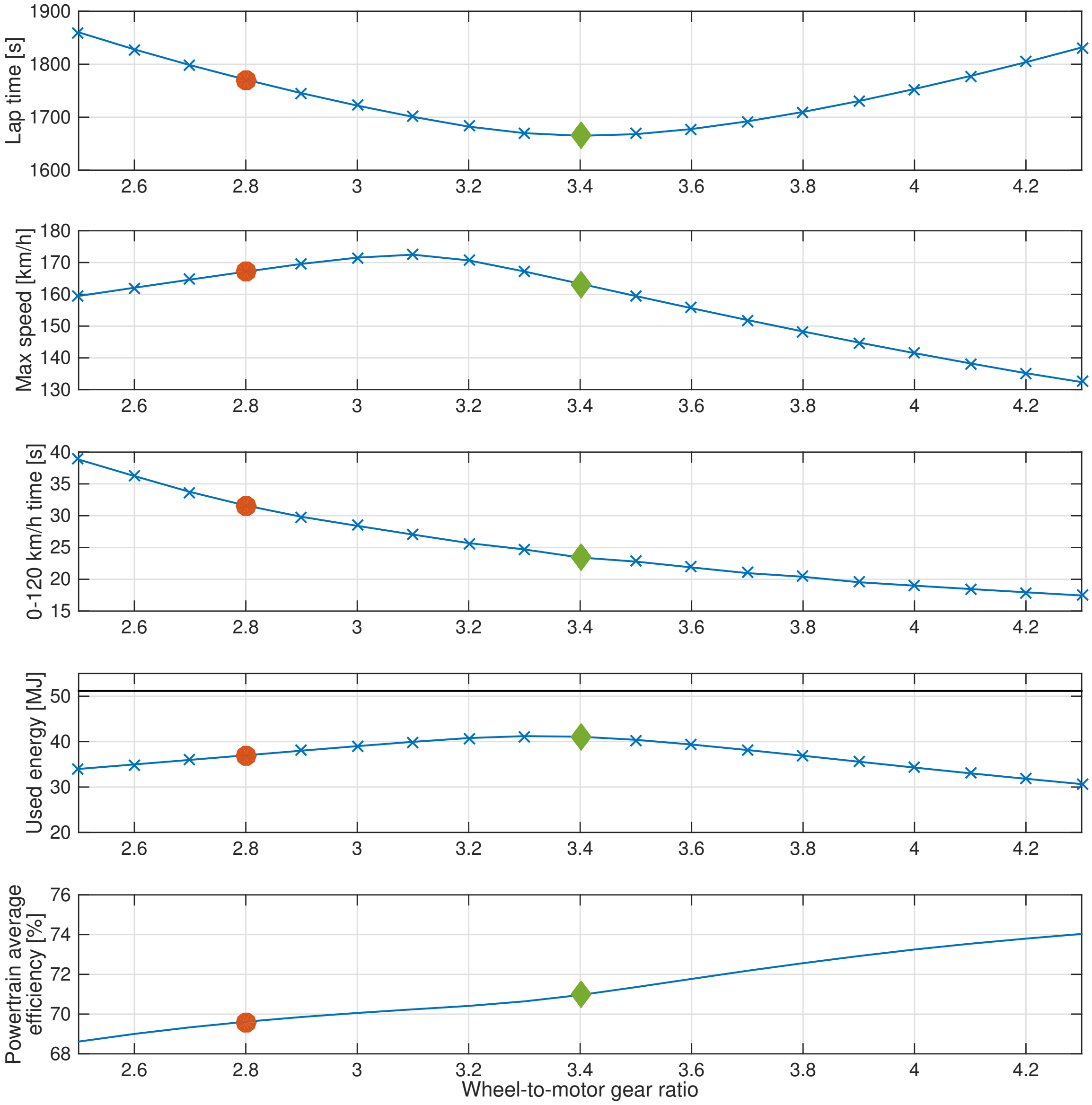


Figure 5.12: From top to bottom, overall lap time, top speed, $0 - 120km/h$ acceleration time, energy used and overall power train efficiency are all shown vs. wheel-to-motor gear ratio. The red circle indicates the configuration used in the TT Zero Challenge, the green diamond indicates the optimal configuration as found through simulation. The black line in the fourth graph from the top represents total energy stored in the battery pack.

the gear ratio, at least in the range considered. The optimal gear ratio 3.4 uses slightly less battery pack energy than gear ratio 3.3, and still allows to diminish the lap time thanks to a higher efficiency.

Motor current, voltage and motorcycle speed are compared in figure 5.13 for the reference

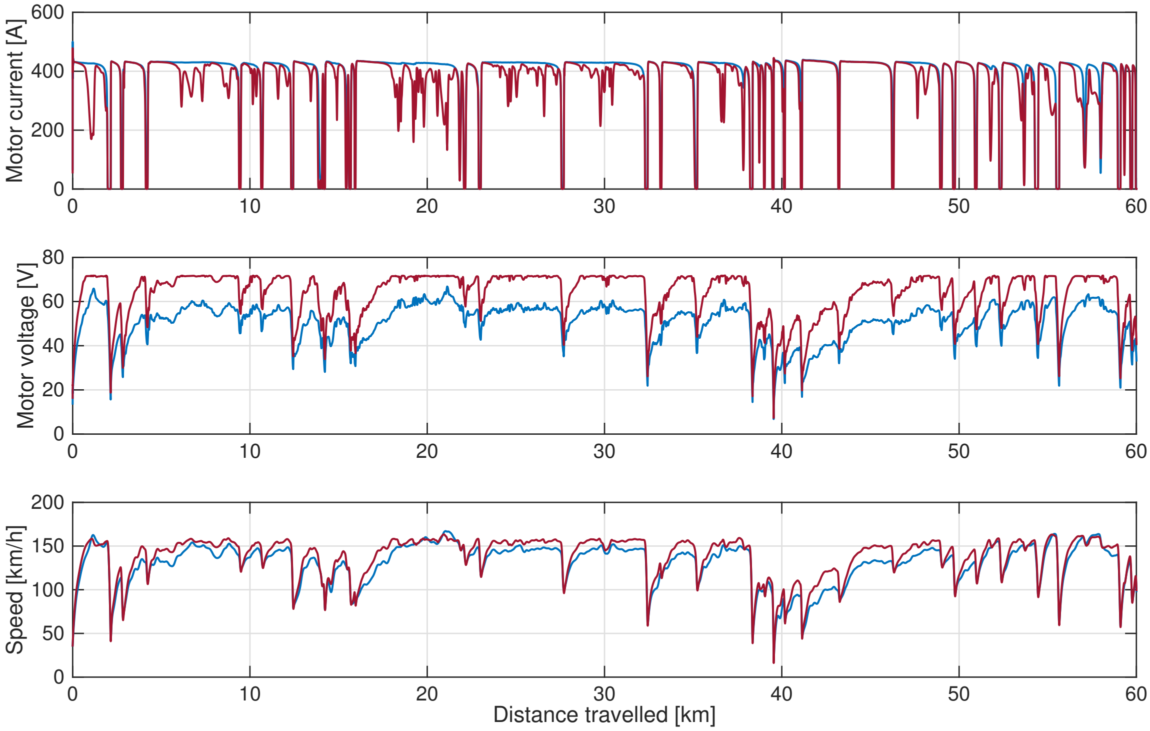


Figure 5.13: Comparison between reference gear ratio 2.8 (blue lines) and optimised gear ratio 3.4 (magenta lines) for motor current, motor voltage and motorcycle speed.

and optimal gear ratios. As expected, the optimal gear ratio uses on average a higher motor driving voltage, but a lower motor current; in particular, with the optimal gear ratio the motor voltage is often equal to the maximum value above which motor damage could occur, while in the reference setup such voltage values are never reached. The speed profile (bottom graph) shows that the optimal gear ratio provides higher accelerations but lower top speed than the reference setup, corresponding to the discussion regarding figure 5.12.

The second parameter under scrutiny is the battery pack capacity: an undersized battery pack will not provide enough power required by the motor, but an oversized battery pack would add extra weight to the motorcycle for the same energy usage. Sizing the pack capacity correctly is therefore again a critical but not straightforward design decision where simulation can help. Several runs haave been simulated with varying battery pack sizes and thus varying amounts of stored energy available, while keeping the energy density constant (at $0.51MJ/kg$ or $142Wh/kg$ as in the experimental reference setup). The results obtained are illustrated in figure 5.14, where the top graph shows lap time and the bottom graph percentage energy used (e.g. 100% means the battery pack has been completely drained). With the lowest battery capacity of $25MJ$ ($6.9kWh$) considered, no energy is left at the end of the race and indeed to conclude the race it is necessary to adopt some energy saving strategies while navigating the course. As pack capacity increases, lap times reduce significantly and minimum lap time is reached for a capacity of (approximately) $37MJ$ ($10.3kWh$). Greater pack capacities are not necessary and the increment in weight reduces performance; however, the extra weight influences the lap time less significantly than energy saving strategies. This suggests that it is more convenient to provide the motorcycle with a slightly over-sized and heavier battery pack, than risking to be forced to adopt an energy conserving riding style.

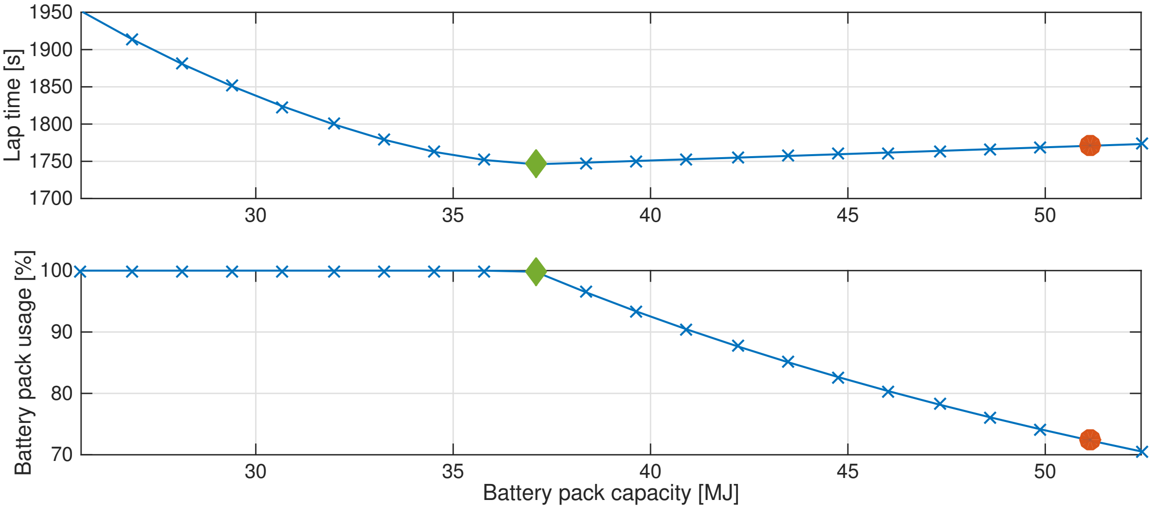


Figure 5.14: Overall lap time (top) and battery usage percentage (bottom) are shown vs. the total amount of energy stored in the battery pack (energy density held constant at reference value $0.51MJ/kg$ or $142Wh/kg$). The red circle indicates the configuration used in the TT Zero Challenge, the green diamond indicates the optimal configuration as found through simulation.

The lap times shown in figure 5.14 are calculated with the reference gear ratio 2.8, which is not optimal; moreover, it has been previously shown that the gear ratio affects the battery usage (figure 5.12). Therefore, it is expected that the gear ratio and battery size should actually be optimised together. Figure 5.15 shows lap time as a function of both gear ratio and battery capacity. A lap time of 27 minutes and 34 seconds (nearly two minutes or $\approx 6.5\%$ faster than the reference configuration) may be obtained with a gear ratio of 3.4 and a battery pack capacity of $\approx 42MJ$ ($11.7kWh$), corresponding to an increase in average motorcycle lap speed from $122km/h$ ($75.8mph$) to $131km/h$ ($81.4mph$). The contour graph highlights that the lap time significantly increases when the chosen gear ratio is far from the optimal one (i.e. greater than 3.6 or lower than 3.3) or when the battery capacity is under-sized (i.e. less than $37MJ$ or $10.3kWh$). Moreover, figure 5.15 confirms what was

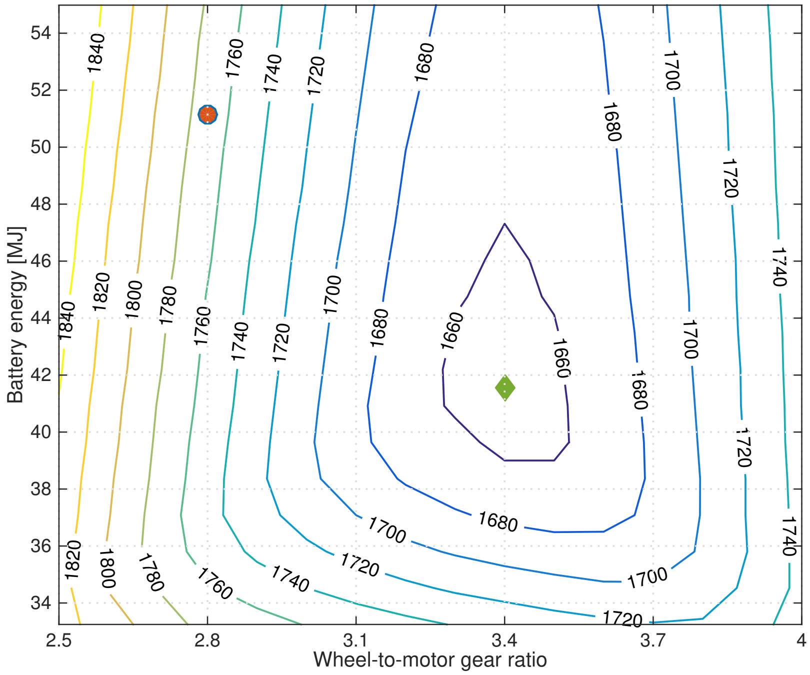


Figure 5.15: Lap time contour graph as a function of gear ratio and battery pack capacity. The minimum lap time of 27 minutes and 34 seconds ($1654s$) is obtained with a gear ratio of 3.4 and a battery pack capacity of $\approx 42MJ$ ($11.7kWh$). The red circle indicates the configuration used in the TT Zero Challenge, the green diamond indicates the optimal configuration as found through simulation.

already observed in figure 5.14, namely that an over-sized battery pack leads to a slight performance decrease, but an under-sized battery pack causes a significant increase of the lap time. By the way the figure shows also that the gear ratio and the battery capacity optimization are almost independent.

The third parameter that has been optimised here through lap time simulation is the number of motors in the power train. BX-15 was equipped with a dual motor configuration, a popular choice because of relative ease of packaging and obvious advantage over a single motor of the same specification. Considering that the performance of the motorcycle studied in this work is limited mainly by motor power, adding one motor seems a worthwhile consideration at first. However one more motor would also add $11kg$ of extra weight, and it would require a bigger battery pack. For this reason it cannot be clear a priori whether two or three motors would constitute the best configuration, and simulation can again provide clarity.

The lap time on the TT course for a motorcycle with a triple motor configuration, with no other changes applied, has been simulated; total mass was increased by $11kg$ (weight of one motor) but the reference gear ratio and battery pack capacity were kept unchanged. The simulated lap time (26 minutes and 6 seconds or $1566s$, as reported in table 5.2) turned out to be significantly lower than that of the dual motor configuration (even of the optimised version). But the simulation showed also that with three motors the 4P24S battery pack is being fully discharged, suggesting that the performance could be increased by parallel mounting an extra battery sequence. However, in 5P24S pack arrangement it turns out that the lap time becomes five seconds more ($1571s$, see table 5.2), therefore the additional energy provided by the extra battery capacity does not compensate for the increased motorcycle weight. In conclusion, a three motor configuration is the best choice in terms of minimum lap time because it allows to increase the average speed on the TT Course by the $\approx 14\%$ from $122km/h$ ($75.8mph$) to $138km/h$ ($85.7mph$) (with the reference gear ratio). However, these results should be considered against the actual design and build process, since a triple motor configuration is more difficult to package and integrate into the motorcycle chassis than the dual motor configuration.

Table 5.2: Comparison of lap times with different motor configurations and battery pack arrangements.

	2 motors & 4P24S ($51MJ$ - $14.2kWh$)	3 motors & 4P24S ($51MJ$ - $14.2kWh$)	3 motors & 5P24S ($64MJ$ - $17.8kWh$)
Lap time	$29m$ $31s$	$26m$ $7s$	$26m$ $11s$
Battery usage	72%	100%	85%

5.8 Summary

Electric and hybrid vehicles are nowadays rapidly gaining in popularity, but performance and energy consumption optimisation remain a challenging task. In this chapter optimal control has been used to simulate an electric motorcycle for the TT Zero Challenge over the entire $60km$-long ($37.73mi$) Snaefell Mountain Course on the Isle of Man. A simple motorcycle multibody model, able to capture the essence of the dynamics, has been used in order to efficiently run simulations over such a long road. A three-dimensional road model and an electric power train model, consisting of battery pack, DC brushed motors and motor controller, have been implemented. The model allows to efficiently perform simulations over the $60km$ road course in less than three minutes, and the simulation outcomes have been validated through comparison with experimental data acquired by the Brunel Racing team during the 2015 TT Zero event.

Lap time simulation was then used to optimise the design of the racing motorcycle, in particular the wheel-to-motor gear ratio, the battery pack capacity and the number of motors in the power train. Results showed that the gear ratio has a significant influence on the performance and on the efficiency of the power train, with the use of the optimal (shorter) gear ratio leading to a lap time reduction of approximately 2 minutes or 6% of the reference experimental data. Perhaps surprisingly, battery pack capacity turned out to affect lap time only marginally, as long as the pack is sized or over-sized so as not to be fully discharged over one lap. However, when battery pack capacity is under-sized, such that energy saving riding strategies must be adopted in order to complete the entire course, the lap time increases significantly as the reduction in motorcycle weight does not overcome the lack of energy to ride at full performance throughout. Finally, the optimisation of the multiple motor configuration demonstrated that adopting an extra motor and upgrading to a triple (from dual) configuration brings along additional performance improvement: the extra available power reduces lap time and overcomes the increase in weight.

These results suggest that the bespoke developed optimal control model for lap time simulation can serve as a valuable tool in the motorcycle design process, because of the sensitivity it harnesses in motorcycle parameter variation studies and the relatively short computing time that is required.

Chapter 6

Direct transcription method for implicit optimal control problems

In the previous chapters the classical formulation of optimal control problem, i.e. where the dynamical system is described by explicit[1] differential equations and the controls are algebraic variables, has been described and applied to minimum lap time problems. In this chapter the formulation including fully implicit differential equations and control derivatives is considered and a direct full collocation transcription method for implicit optimal control problems is presented. The aim is to provide a tool for optimal control problems that leave a larger freedom from a modelling point of view. In the next section, the advantages of implicit equations is described, then the implicit optimal control formulation is presented and the first order necessary conditions are derived. The direct transcription method for solving the implicit OCP is then described and implemented in the *Maverick* software that is currently in use at the University of Padova. Finally, the implicit OCP formulation is applied on a case-of-study problem of a motorbike performing a U-turn manoeuvre on a three-dimensional road.

6.1 Advantages of the implicit formulation

Optimal control theory, at least in the form it can be found in classic books as [76, 20, 75, 11] has always considered OCPs with explicit first order equations, i.e. where the dynamical system of the optimal control problem is described by differential equations in the form:

$$\boldsymbol{x}' = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{p}, \zeta) \tag{6.1}$$

[1] In the previous chapters, when using the *Pins* software, equations of motion have always been considered also implicit but linear in the derivative, i.e. they can be made explicit with a matrix inversion.

Moreover, many commercial numerical optimal control software have been developed so as to deal only with explicit first order equations (6.1), like *GPOPS-II* and *Falcon*. Explicit first order equations however may represent a limiting factor in some optimal control problem applications. For instance, whenever the dynamical system of the optimal control problem is a mechanical or multibody system, the first order equations (2.6c) arise from the equations of motion, and the state variable vector $\boldsymbol{x}$ is strictly related to the degrees of freedom of the system. For a multibody system the first-order equations, that are obtained by reducing to the first order the equations of motion (that are of the second order), can be generally written in the form:

$$\boldsymbol{A}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{p}, \zeta)\boldsymbol{x}' = \boldsymbol{B}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{p}, \zeta) \tag{6.2}$$

where $\boldsymbol{A}$ is a square matrix (called mass matrix) and $\boldsymbol{B}$ a vector. However, equation (6.2) cannot be used as dynamical equation (2.6c) in OCPs because the classic OCP theory do not apply to such equation. If the mass matrix is diagonal or if its inverse can be exactly calculated, equation (6.2) can be put in the explicit form (6.1). Usually, for simple systems with few degrees of freedom, i.e. up to approximately ten state variables, the exact (not numerical) inverse of the mass matrix can be calculated. However optimal control problems may include more state variables so as that the mass matrix inverse can be calculated only numerically; an example are the equations of the motorbike multibody model used in [85, 28, 27] or of the GP2 car model described in section 4.3. When the mass matrix cannot be exactly inverted, an explicit optimal control software can still be used if "tricky" strategies are adopted. Two possibilities, for instance, are the following:

- the mass matrix is inverted numerically and the first order equations are calculated as $\boldsymbol{x}' = \boldsymbol{A}^{-1}\boldsymbol{B}$, where the matrix $\boldsymbol{A}$ and the vector $\boldsymbol{B}$ are numerically evaluated. This strategy has two main drawbacks: first, the evaluation of the equations is slower because it requires a matrix inversion and, second, the Jacobian and Hessian matrices for the equations are usually numerically calculated. It is known that the use of numerically approximated derivatives makes the solving algorithm slower and often less robust [20, 11].

- algebraic variables $\boldsymbol{a}^*$ are introduced into the optimal control problem, so as that equations 6.2 are replaced by $\boldsymbol{x}' = \boldsymbol{a}^*$. Moreover, the algebraic variables must satisfy the path constraints $\boldsymbol{A}\boldsymbol{a}^* = \boldsymbol{B}$. The explicit form of the equations of motion is thus obtained at the expense of additional controls and path constraints, resulting into a problem of significantly greater size. For some problems the implicit form is more robust and the solver converge faster to solution [31].

Despite the equations of motion of all multibody systems can be — at least in theory —

written in the linear-implicit form (6.2), in the case of complex dynamical systems the equations may not be straightforwardly written as in (6.2). Moreover, some dynamical systems may be described by equations which are intrinsically implicit in the state derivatives[2]; an example is the muscular activation dynamics problem treated in [31]. In such problem the equations cannot be written neither in explicit form, nor in the linear-implicit one (6.2).

From a more general point of view, every first order dynamical system is described by differential equations in the form:

$$\boldsymbol{f}(\boldsymbol{x}', \boldsymbol{x}, \boldsymbol{u}, \boldsymbol{p}, \zeta) = 0 \tag{6.3}$$

The availability of a numerical optimal control solver that allows to use implicit equations as in (6.3) is of great advantage from the modelling point of view, especially when the dynamical system equations are cumbersome to write as in (6.1) or (6.3). However, if an OCP is governed by an implicit dynamical equation (6.3), the classical OCP theory does not guarantee the existence of a solution. In the next section some sufficient conditions for the existence of a solution are discussed.

6.2 Implicit optimal control formulation

An implicit optimal control problem is characterised by implicit first order equations:

$$\boldsymbol{f}(\boldsymbol{x}', \boldsymbol{x}, \boldsymbol{u}, \boldsymbol{p}, \zeta) = 0 \tag{6.4}$$

Equation (6.4) represents a general DAE system where part of the state variables $\boldsymbol{x}$ may be algebraic, i.e. they are determined by algebraic equations and their derivatives do not appear. In classical optimal control formulations, controls $\boldsymbol{u}$ have been always considered piecewise continuous algebraic variables, and their derivatives do not appear in the OCP. However, under these hypothesis, an OCP described by equation (6.4) is not guaranteed to have one, and only one, solution. Indeed, the Pontryagin Maximum Principle cannot be applied here. If calculus of variation is exploited to obtain the necessary conditions for the solution, then the derivative $\frac{d}{d\zeta}\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}'}$ has to be calculated, thus the control and all other algebraic variables have to be differentiable (unless the equations have a particular structure, but this depends on the specific dynamical system). The exact calculus of the necessary conditions through CV will be carried out in the next section. It is opinion of the author that such prerequisites for the controls and the algebraic variables (i.e. to be differentiable) are probably not the most general hypothesis under which the existence and

[2] In such dynamical system the control is by definition differentiable, while in classic OCP theory it is required to be only piecewise continuous, as discussed in section 2.3.1.2.

uniqueness of a solution for an OCP described by the implicit dynamical equation (6.4) is guaranteed, exactly as, in classical OCP theory, CV has more stringent hypothesis than the PMP. The development of a complete and solid theory for implicit OCPs is out of the scope of this chapter, as it would require an entire thesis dedicated to this subject.

At first sight it may seem a very strict requirement for the controls to be differentiable, since there are plenty of classic OCP examples where the controls are discontinuous. In particular, in all OCPs where the dynamical equations and the target are linear in the controls, the solution exhibits a bang-bang behaviour (where there are no singular arcs), i.e. the controls jump discontinuously between their maximum and minimum values. However, even if a relatively small regularisation term proportional to the square of the controls is added to the Lagrange target, the theoretical bang-bang behaviour is lost and the controls do not show discontinuities. This regularisation technique is often used when solving numerically OCPs to make the problem less stiff and easier to solve. Moreover, when the penalty approach (see section 2.3.1.3) is adopted as solving strategy, the penalties act as regularising terms and the OCP solution presents smooth controls. For these reasons, it is opinion of the author that from a practical point of view, the requirement of differentiable controls is not too restrictive since OCPs whose solution is characterised by discontinuous controls can be regularised to obtain approximated solutions with smooth controls. Moreover, in the real world it is often impossible to reproduce discontinuous (in the mathematical sense) controls.

If controls are assumed to be differentiable, it can be useful to use their derivatives in the OCP formulation, i.e. in the target and constraints expressions. This gives a greater flexibility in the OCP modelling: regularisation terms related to the control derivatives can be included into the target so as to suppress control oscillations, and bounds on the control rates can be imposed as path constraints. Therefore it comes natural to let the first order equations (6.4) — and similarly the OCP target and constraints — depend on the control derivatives. This requires the controls to be twice differentiable in order to use CV to derive the necessary conditions.

Generalising further the OCP formulation, the optimal control problem can be multiphase, i.e. the first order equations, target and constraints may change (a fixed number of times) in each phase. A classical example is the launch of a multi-stage rocket. The resulting formulation for a multi-phase ($p = 1, \ldots, P$) implicit optimal control problem, with differential controls, is the following:

$$\min_{\boldsymbol{u}} \sum_{p=1}^{P} \psi^{(p)}(\boldsymbol{x}(\zeta_i^{(p)}), \boldsymbol{x}(\zeta_f^{(p)}), \boldsymbol{u}(\zeta_i^{(p)}), \boldsymbol{u}(\zeta_f^{(p)}), \boldsymbol{p}, \zeta_i^{(p)}, \zeta_f^{(p)}) \tag{6.5a}$$

$$+ \int_{\zeta_i^{(p)}}^{\zeta_f^{(p)}} \ell^{(p)}(\boldsymbol{x}(\zeta), \boldsymbol{x}'(\zeta), \boldsymbol{u}(\zeta), \boldsymbol{u}'(\zeta), \boldsymbol{p}, \zeta) d\zeta \tag{6.5b}$$

$$\text{subject to: (for } p = 1, \ldots, P) \tag{6.5c}$$

$$\boldsymbol{f}^{(p)}(\boldsymbol{x}(\zeta), \boldsymbol{x}'(\zeta), \boldsymbol{u}(\zeta), \boldsymbol{u}'(\zeta), \boldsymbol{p}, \zeta) = 0 \tag{6.5d}$$

$$\boldsymbol{g}_l^{(p)} \leq \boldsymbol{g}^{(p)}(\boldsymbol{x}(\zeta), \boldsymbol{x}'(\zeta), \boldsymbol{u}(\zeta), \boldsymbol{u}'(\zeta), \boldsymbol{p}, \zeta) \leq \boldsymbol{g}_u^{(p)} \tag{6.5e}$$

$$\boldsymbol{h}_l^{(p)} \leq \int_{\zeta_i^{(p)}}^{\zeta_f^{(p)}} \boldsymbol{h}^{(p)}(\boldsymbol{x}(\zeta), \boldsymbol{x}'(\zeta), \boldsymbol{u}(\zeta), \boldsymbol{u}'(\zeta), \boldsymbol{p}, \zeta) d\zeta \leq \boldsymbol{h}_u^{(p)} \tag{6.5f}$$

$$\boldsymbol{\phi}_l^{(p)} \leq \boldsymbol{\phi}^{(p)}(\boldsymbol{x}(\zeta_i^{(p)}), \boldsymbol{x}(\zeta_f^{(p)}), \boldsymbol{u}(\zeta_i^{(p)}), \boldsymbol{u}(\zeta_f^{(p)}), \boldsymbol{p}, \zeta_i^{(p)}, \zeta_f^{(p)}) \leq \boldsymbol{\phi}_u^{(p)} \tag{6.5g}$$

$$\boldsymbol{x}_l^{(p)} \leq \boldsymbol{x}^{(p)} \leq \boldsymbol{x}_u^{(p)} \tag{6.5h}$$

$$\boldsymbol{u}_l^{(p)} \leq \boldsymbol{u}^{(p)} \leq \boldsymbol{u}_u^{(p)} \tag{6.5i}$$

$$\boldsymbol{p}_l^{(p)} \leq \boldsymbol{p}^{(p)} \leq \boldsymbol{p}_u^{(p)} \tag{6.5j}$$

$$\boldsymbol{\alpha}_l \leq \boldsymbol{\alpha}(\boldsymbol{x}(\zeta_i^{1,\ldots,P}), \boldsymbol{x}(\zeta_f^{1,\ldots,P}), \boldsymbol{u}(\zeta_i^{1,\ldots,P}), \boldsymbol{u}(\zeta_f^{1,\ldots,P}), \boldsymbol{p}, \zeta_i^{1,\ldots,P}, \zeta_f^{1,\ldots,P}) \leq \boldsymbol{\alpha}_u \tag{6.5k}$$

where the subscripts $_{l,u}$ indicate respectively the minimum (lower) and maximum (upper) bounds and the superscript $^{(p)}$ denotes the phase. Since controls are assumed differentiable, controls and states are completely equivalent from a mathematical point of view; indeed, it can be noted that the above reported formulation (6.5) is symmetric between states and controls. The only distinction between states and controls is that the latter are the inputs of the dynamical system and the former are determined by the control inputs.

6.2.1 First order necessary conditions

In this section the first order necessary conditions for the implicit OCP (6.5) are calculated. Such necessary conditions give the equations that the Lagrange multipliers satisfy, which will be used in the next sections to find the relationship between the Lagrange multipliers and the KKT multipliers of the direct transcription method that will be presented.

For sake of conciseness, a single-phase version of the problem (6.5) is here considered; the extension to the multi-phase problem is straightforward. Moreover the constraints on the states, controls and parameters (6.5h) (6.5i) (6.5j) will be neglected since they can be thought as included in the path constraints (6.5e).

The first order necessary conditions are different from those derived in chapter 2 since the new problem has a different formulation. One of the differences is that in the implicit problem the target (6.5b), the first order equations (6.5d) and the constraints (6.5e) (6.5f) depend on the control derivatives. Moreover, the boundary conditions now depend on the controls and the first order equations have an different structure. While this does not prevent to use CV to derive the necessary conditions[3], problem (6.5) presents some inequality constraints which makes it unsuitable for CV. Thus, the necessary conditions can be calculated only if constraints (6.5e), (6.5f) and (6.5g) are replaced by the following equality constraints:

$$g_j^*(\boldsymbol{x}(\zeta), \boldsymbol{x}'(\zeta), \boldsymbol{u}(\zeta), \boldsymbol{u}'(\zeta), \boldsymbol{p}, \zeta) = g_j^*(\zeta) \tag{6.6a}$$

$$\int_{\zeta_i}^{\zeta_f} \boldsymbol{h}_j^*(\boldsymbol{x}(\zeta), \boldsymbol{x}'(\zeta), \boldsymbol{u}(\zeta), \boldsymbol{u}'(\zeta), \boldsymbol{p}, \zeta) d\zeta = h_j^* \tag{6.6b}$$

$$\phi_j(\boldsymbol{x}(\zeta_i^{(p)}), \boldsymbol{x}(\zeta_f^{(p)}), \boldsymbol{u}(\zeta_i), \boldsymbol{u}(\zeta_f), \boldsymbol{p}, \zeta_i, \zeta_f) = \hat{\phi}_j^* \tag{6.6c}$$

where the superscript * indicates the optimum value, i.e. the value assumed by the inequality constraints along the solution. Even if inequality constraints are replaced by equality ones, since $g_j^*(\zeta)$, h_j^*, ϕ_j^*, are the values at the solution, the Lagrange multipliers remain unchanged, thus they are zero unless when the inequality constraints are active. Clearly, the optimum values $g_j^*(\zeta)$, h_j^* and ϕ_j^* cannot be known before the solution is calculated, so the necessary conditions cannot be used to really find the solution. However, it is possible to virtually substitute the inequality constraints with equations (6.6) just to derive the equations that the Lagrange multipliers satisfy.

The necessary conditions for the equality constrained problem are now calculated. The Lagrangian of the problem is:

$$\begin{aligned}
\hat{J} = &\boldsymbol{\psi}(\boldsymbol{x}(\zeta_i), \boldsymbol{x}(\zeta_f), \boldsymbol{u}(\zeta_i), \boldsymbol{u}(\zeta_f), \boldsymbol{p}, \zeta_i, \zeta_f) + \boldsymbol{\gamma}^\mathsf{T} \boldsymbol{\phi}(\boldsymbol{x}(\zeta_i), \boldsymbol{x}(\zeta_f), \boldsymbol{u}(\zeta_i), \boldsymbol{u}(\zeta_f), \boldsymbol{p}, \zeta_i, \zeta_f) \\
&+ \int_{\zeta_i}^{\zeta_f} \left[\ell(\boldsymbol{x}, \boldsymbol{x}', \boldsymbol{u}, \boldsymbol{u}', \boldsymbol{p}, \zeta) + \boldsymbol{\lambda}(\zeta)^\mathsf{T} \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{x}', \boldsymbol{u}, \boldsymbol{u}', \boldsymbol{p}, \zeta) \right] d\zeta \\
&+ \int_{\zeta_i}^{\zeta_f} \boldsymbol{\omega}(\zeta)^\mathsf{T} \left(\boldsymbol{g}\left(\boldsymbol{x}, \boldsymbol{x}', \boldsymbol{u}, \boldsymbol{u}', \boldsymbol{p}, \zeta\right) - \boldsymbol{g}^* \right) d\zeta \\
&+ \boldsymbol{\upsilon}^\mathsf{T} \left(\int_{\zeta_i}^{\zeta_f} \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{x}', \boldsymbol{u}, \boldsymbol{u}', \boldsymbol{p}, \zeta) d\zeta - \boldsymbol{h}^* \right)
\end{aligned} \tag{6.7}$$

where $\boldsymbol{\gamma}$, $\boldsymbol{\lambda}(\zeta)$, $\boldsymbol{\omega}(\zeta)$ and $\boldsymbol{\upsilon}$ are the Lagrange multipliers. As previously said, it follows from the definition of g_j^*, h_j^*, ϕ_j^* that these Lagrange multipliers are zero when the constraints

[3] Under the hypothesis that states and controls are twice differentiable.

are not active:

$$\begin{aligned} \omega_j(\zeta) &= 0 \qquad \text{if } \zeta \notin \boldsymbol{A}_{g,j} \\ \upsilon_j &= 0 \qquad \text{if } j \notin \boldsymbol{A}_h \\ \gamma_j &= 0 \qquad \text{if } j \notin \boldsymbol{A}_\phi \end{aligned} \tag{6.8}$$

where the sets $\boldsymbol{A}_{g,j}$, $\boldsymbol{A}_h$, $\boldsymbol{A}_\phi$ are the active constraints set:

$$\begin{aligned} \hat{\zeta} \in \boldsymbol{A}_{g,j} &\iff g_j \text{ is active at } \zeta = \hat{\zeta} \\ j \in \boldsymbol{A}_h &\iff h_j \text{ is active} \\ j \in \boldsymbol{A}_\phi &\iff \phi_j \text{ is active} \end{aligned} \tag{6.9}$$

An inequality constraints is said to be active if its value is equal to its lower or upper bound.

Taking the first variation of the unconstrained target $\hat{J}$ (6.7) and equating it to zero for any arbitrary variation in all the variables and multipliers, the following equations (first order necessary conditions) are obtained:

$$\mathbf{0} = \frac{\partial \psi}{\partial \boldsymbol{v}}\Big|_{\zeta_i} + \boldsymbol{\gamma}^\mathsf{T} \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{v}}\Big|_{\zeta_i} - \frac{\partial \ell}{\partial \boldsymbol{v}'}\Big|_{\zeta_i} - \boldsymbol{\lambda}^\mathsf{T} \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{v}'}\Big|_{\zeta_i} - \boldsymbol{\omega}^\mathsf{T} \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}'}\Big|_{\zeta_i} - \boldsymbol{\upsilon}^\mathsf{T} \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}'}\Big|_{\zeta_i} \tag{6.10a}$$

$$\mathbf{0} = \frac{\partial \psi}{\partial \boldsymbol{v}}\Big|_{\zeta_i} + \boldsymbol{\gamma}^\mathsf{T} \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{v}}\Big|_{\zeta_f} + \frac{\partial \ell}{\partial \boldsymbol{v}'}\Big|_{\zeta_f} + \boldsymbol{\lambda}^\mathsf{T} \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{v}'}\Big|_{\zeta_f} + \boldsymbol{\omega}^\mathsf{T} \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}'}\Big|_{\zeta_f} + \boldsymbol{\upsilon}^\mathsf{T} \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}'}\Big|_{\zeta_f} \tag{6.10b}$$

$$\mathbf{0} = \frac{\partial \ell}{\partial \boldsymbol{v}} + \boldsymbol{\lambda}^\mathsf{T} \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{v}} + \boldsymbol{\omega}^\mathsf{T} \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{v}} + \boldsymbol{\upsilon}^\mathsf{T} \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{v}} + \frac{d}{d\zeta} \left(\frac{\partial \ell}{\partial \boldsymbol{v}'} + \boldsymbol{\lambda}^\mathsf{T} \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{v}'} + \boldsymbol{\omega}^\mathsf{T} \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{v}'} + \boldsymbol{\upsilon}^\mathsf{T} \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{v}'} \right) \tag{6.10c}$$

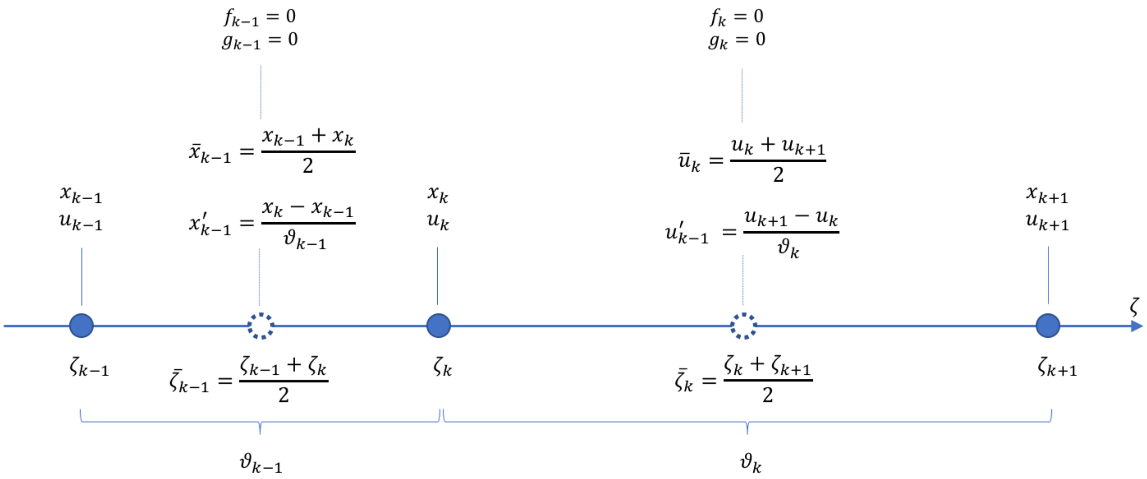
$$\mathbf{0} = \frac{\partial \psi}{\partial \boldsymbol{p}} + \boldsymbol{\gamma}^\mathsf{T} \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{p}} + \int_{\zeta_i}^{\zeta_f} \left(\frac{\partial \ell}{\partial \boldsymbol{p}} + \boldsymbol{\lambda}^\mathsf{T} \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{p}} + \boldsymbol{\omega}^\mathsf{T} \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{p}} + \boldsymbol{\upsilon}^\mathsf{T} \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{p}} \right) d\zeta \tag{6.10d}$$

$$\mathbf{0} = \boldsymbol{\phi}; \quad \mathbf{0} = \boldsymbol{f}; \quad \mathbf{0} = \boldsymbol{g} - \boldsymbol{g}^*; \quad \mathbf{0} = \int_{\zeta_i}^{\zeta_f} \boldsymbol{h} \; d\zeta - \boldsymbol{h}^* \tag{6.10e}$$

Equations (6.10e) simply express the constraints satisfaction (including the first order equations). These equations are originated from the first order variation of the unconstrained target with respect to the Lagrange multipliers. Since the Lagrange multipliers $\boldsymbol{\omega}$, $\boldsymbol{\gamma}$, $\boldsymbol{\upsilon}$ are identically zero outside the active sets (6.9), equations (6.10e) can be eventually imposed only inside the active sets, i.e.:

$$\begin{aligned} 0 &= \phi_j && \text{for } \phi_j \in \boldsymbol{A}_\phi \\ 0 &= g_j - g_j^* && \text{for } \zeta \in \boldsymbol{A}_{g,j} \\ 0 &= \int_{\zeta_i}^{\zeta_f} h_j d\zeta - h_j^* && \text{for } h_j \in \boldsymbol{A}_h \end{aligned} \tag{6.11}$$

Equations (6.10a) to (6.10d) are instead the co-equations that determine the Lagrange multipliers.



$f_{k-1} = 0$
$g_{k-1} = 0$
$\bar{x}_{k-1} = \frac{x_{k-1} + x_k}{2}$
$x'_{k-1} = \frac{x_k - x_{k-1}}{\vartheta_{k-1}}$
$f_k = 0$
$g_k = 0$
$\bar{u}_k = \frac{u_k + u_{k+1}}{2}$
$u'_{k-1} = \frac{u_{k+1} - u_k}{\vartheta_k}$
x_{k-1}
u_{k-1}
x_k
u_k
x_{k+1}
u_{k+1}
ζ
ζ_{k-1}
$\bar{\zeta}_{k-1} = \frac{\zeta_{k-1} + \zeta_k}{2}$
ζ_k
$\bar{\zeta}_k = \frac{\zeta_k + \zeta_{k+1}}{2}$
ζ_{k+1}
ϑ_{k-1}
ϑ_k

Figure 6.1: Midpoint integration scheme used for the transcription of the OCP into an NLP. The states $\boldsymbol{x}$ and controls $\boldsymbol{u}$ are discretised on each mesh point, equations and constraints are evaluated at the middle of each mesh interval.

Summarising, the first order necessary conditions for the implicit OCP (6.6) are equations (6.10), where the last equations (6.10e) can be eventually replaced by (6.11).

6.3 Direct full collocation to NLP transcription

In this section a direct full collocation method for solving the implicit optimal control problem (6.5) is described. First, the chosen discretisation scheme is presented and then resulting NLP problem structure is described. The close relationship between the NLP Lagrange multipliers and the Lagrange multipliers of the first order necessary conditions (6.10) is then derived. Finally, a brief overview of the software implementation is presented.

6.3.1 Discretisation scheme

Every numerical approach to optimal control problem, regardless it is indirect or direct, relies upon a discretisation scheme. The discretisation scheme allows to transform the infinite dimensional problem (6.5) into a finite dimensional one that can be numerically solved using a calculator. For the direct full collocation transcription described in this section, a midpoint discretisation scheme has been chosen. Even though it is not as accurate as Gaussian methods [88, 77], it has the advantage that it leads to sparser Jacobian and Hessian matrices. Moreover, being an implicit method, it does not suffer of high instability issued as the explicit ones, like the forward Euler method.

In the description of the discretisation scheme, a single-phase ($P = 1$) problem is considered for sake of conciseness. The extension to a multi-phase one is straightforward. Considering problem (6.5), the independent variable ζ is discretised into $N+1$ mesh points $\zeta_0, \zeta_1, \dots, \zeta_N$, with $\zeta_0 = \zeta_i$ and $\zeta_N = \zeta_f$. Therefore the interval $(\zeta_i,\ \zeta_f)$ is divided into N mesh intervals of length $\theta_k = \zeta_{k+1} - \zeta_k$ for $k = 1, \dots, N$, as depicted in figure 6.1. The middle point of each mesh interval are indicated with an over-line $\overline{\zeta}_k = (\zeta_{k+1} - \zeta_k)/2$. The state variables $\boldsymbol{x}$ and controls $\boldsymbol{u}$ are discretised in correspondence of the mesh points $\boldsymbol{x}_k = \boldsymbol{x}(\zeta_k)$ and $\boldsymbol{u}_k = \boldsymbol{u}(\zeta_k)$ (see figure 6.1). On the contrary, the parameters $\boldsymbol{p}$ does not require any discretisation since they are already discrete variables.

In order to evaluate the first order equations (6.5d) at the middle of each mesh interval, an estimate of the state and control and of their derivatives is required. The state and control derivatives are estimated at the middle of each mesh interval $\overline{\zeta}_k$ accordingly to the midpoint method:

$$\begin{aligned} \boldsymbol{x}'_k &\equiv \left.\frac{d\boldsymbol{x}}{d\zeta}\right|_{\overline{\zeta}_k} \approx \frac{\boldsymbol{x}_{k+1} - \boldsymbol{x}_k}{\theta_k} \\ \boldsymbol{u}'_k &\equiv \left.\frac{d\boldsymbol{u}}{d\zeta}\right|_{\overline{\zeta}_k} \approx \frac{\boldsymbol{u}_{k+1} - \boldsymbol{u}_k}{\theta_k} \end{aligned} \tag{6.12}$$

Similarly, the state and control at the middle of each mesh interval are estimated as the average of their values at the interval boundaries:

$$\begin{aligned} \overline{\boldsymbol{x}}_k &\equiv \boldsymbol{x}(\overline{\zeta}_k) \approx \frac{\boldsymbol{x}_{k+1} + \boldsymbol{x}_k}{2} \\ \overline{\boldsymbol{u}}_k &\equiv \boldsymbol{u}(\overline{\zeta}_k) \approx \frac{\boldsymbol{u}_{k+1} - \boldsymbol{u}_k}{2} \end{aligned} \tag{6.13}$$

The Lagrange target (6.5b), first order equations (6.5d), path (6.5e) and integral (6.5f) constraints are thus evaluated at the N points $\overline{\zeta}_0, \dots, \overline{\zeta}_N$:

$$\ell(\boldsymbol{x}(\overline{\zeta}_k), \boldsymbol{x}'(\overline{\zeta}_k), \boldsymbol{u}(\overline{\zeta}_k), \boldsymbol{u}'(\overline{\zeta}_k), \boldsymbol{p}, \overline{\zeta}_k) \approx \ell(\overline{\boldsymbol{x}}_k, \boldsymbol{x}'_k, \overline{\boldsymbol{u}}_k, \boldsymbol{u}'_k, \boldsymbol{p}, \overline{\zeta}_k) \qquad = \ell_k \tag{6.14a}$$

$$\boldsymbol{f}(\boldsymbol{x}(\overline{\zeta}_k), \boldsymbol{x}'(\overline{\zeta}_k), \boldsymbol{u}(\overline{\zeta}_k), \boldsymbol{u}'(\overline{\zeta}_k), \boldsymbol{p}, \overline{\zeta}_k) \approx \boldsymbol{f}(\overline{\boldsymbol{x}}_k, \boldsymbol{x}'_k, \overline{\boldsymbol{u}}_k, \boldsymbol{u}'_k, \boldsymbol{p}, \overline{\zeta}_k) \qquad = \boldsymbol{f}_k \tag{6.14b}$$

$$\boldsymbol{g}(\boldsymbol{x}(\overline{\zeta}_k), \boldsymbol{x}'(\overline{\zeta}_k), \boldsymbol{u}(\overline{\zeta}_k), \boldsymbol{u}'(\overline{\zeta}_k), \boldsymbol{p}, \overline{\zeta}_k) \approx \boldsymbol{g}(\overline{\boldsymbol{x}}_k, \boldsymbol{x}'_k, \overline{\boldsymbol{u}}_k, \boldsymbol{u}'_k, \boldsymbol{p}, \overline{\zeta}_k) \qquad = \boldsymbol{g}_k \tag{6.14c}$$

$$\boldsymbol{h}(\boldsymbol{x}(\overline{\zeta}_k), \boldsymbol{x}'(\overline{\zeta}_k), \boldsymbol{u}(\overline{\zeta}_k), \boldsymbol{u}'(\overline{\zeta}_k), \boldsymbol{p}, \overline{\zeta}_k) \approx \boldsymbol{h}(\overline{\boldsymbol{x}}_k, \boldsymbol{x}'_k, \overline{\boldsymbol{u}}_k, \boldsymbol{u}'_k, \boldsymbol{p}, \overline{\zeta}_k) \qquad = \boldsymbol{h}_k \tag{6.14d}$$

Finally, the Mayer target (6.5a) and the boundary conditions (6.5g) are evaluated at the domain extrema ζ_0 and ζ_N:

$$\psi(\boldsymbol{x}(\zeta_i), \boldsymbol{x}(\zeta_f), \boldsymbol{u}(\zeta_i), \boldsymbol{u}(\zeta_f), \boldsymbol{p}, \zeta_i, \zeta_f) = \psi(\boldsymbol{x}_0, \boldsymbol{x}_N, \boldsymbol{u}_0, \boldsymbol{u}_N, \boldsymbol{p}, \zeta_0, \zeta_N) \tag{6.15a}$$

$$\boldsymbol{\phi}(\boldsymbol{x}(\zeta_i), \boldsymbol{x}(\zeta_f), \boldsymbol{u}(\zeta_i), \boldsymbol{u}(\zeta_f), \boldsymbol{p}, \zeta_i, \zeta_f) = \boldsymbol{\phi}(\boldsymbol{x}_0, \boldsymbol{x}_N, \boldsymbol{u}_0, \boldsymbol{u}_N, \boldsymbol{p}, \zeta_0, \zeta_N) \tag{6.15b}$$

With the midpoint discretisation scheme here adopted, derivatives have a very sparse structure. Indeed, given the discrete equations $\boldsymbol{f}_k$ and constraints $\boldsymbol{g}_k$, $\boldsymbol{h}_k$, the following relationships hold for their derivatives:

$$\begin{aligned}
\frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{x}_k} &= \frac{1}{2}\frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{x}} - \frac{1}{\theta_k}\frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{x}'}, & \frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{x}_{k+1}} &= \frac{1}{2}\frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{x}} + \frac{1}{\theta_k}\frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{x}'}, & \frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{x}_j} &= \mathbf{0} \text{ for } j \neq k, k+1 \\
\frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{u}_k} &= \frac{1}{2}\frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{u}} - \frac{1}{\theta_k}\frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{u}'}, & \frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{u}_{k+1}} &= \frac{1}{2}\frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{u}} + \frac{1}{\theta_k}\frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{u}'}, & \frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{u}_j} &= \mathbf{0} \text{ for } j \neq k, k+1 \\
\frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{x}_k} &= \frac{1}{2}\frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{x}} - \frac{1}{\theta_k}\frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{x}'}, & \frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{x}_{k+1}} &= \frac{1}{2}\frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{x}} + \frac{1}{\theta_k}\frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{x}'}, & \frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{x}_j} &= \mathbf{0} \text{ for } j \neq k, k+1 \\
\frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{u}_k} &= \frac{1}{2}\frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{u}} - \frac{1}{\theta_k}\frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{u}'}, & \frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{u}_{k+1}} &= \frac{1}{2}\frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{u}} + \frac{1}{\theta_k}\frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{u}'}, & \frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{u}_j} &= \mathbf{0} \text{ for } j \neq k, k+1 \qquad (6.16) \\
\frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{x}_k} &= \frac{1}{2}\frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{x}} - \frac{1}{\theta_k}\frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{x}'}, & \frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{x}_{k+1}} &= \frac{1}{2}\frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{x}} + \frac{1}{\theta_k}\frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{x}'}, & \frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{x}_j} &= \mathbf{0} \text{ for } j \neq k, k+1 \\
\frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{u}_k} &= \frac{1}{2}\frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{u}} - \frac{1}{\theta_k}\frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{u}'}, & \frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{u}_{k+1}} &= \frac{1}{2}\frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{u}} + \frac{1}{\theta_k}\frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{u}'}, & \frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{u}_j} &= \mathbf{0} \text{ for } j \neq k, k+1 \\
\frac{\partial \ell_k}{\partial \boldsymbol{u}_k} &= \frac{1}{2}\frac{\partial \ell_k}{\partial \boldsymbol{u}} - \frac{1}{\theta_k}\frac{\partial \ell_k}{\partial \boldsymbol{u}'}, & \frac{\partial \ell_k}{\partial \boldsymbol{u}_{k+1}} &= \frac{1}{2}\frac{\partial \ell_k}{\partial \boldsymbol{u}} + \frac{1}{\theta_k}\frac{\partial \ell_k}{\partial \boldsymbol{u}'}, & \frac{\partial \ell_k}{\partial \boldsymbol{u}_j} &= \mathbf{0} \text{ for } j \neq k, k+1
\end{aligned}$$

The above equations state that each NLP constraint, regardless it is $\boldsymbol{f}_k$, $\boldsymbol{g}_k$, $\boldsymbol{h}_k$, depends only on few NLP variables: $\boldsymbol{x}_k$, $\boldsymbol{x}_{k+1}$, $\boldsymbol{u}_k$, $\boldsymbol{u}_{k+1}$. Moreover the same holds for the discretised target ℓ_k. These relationship will be extensively used when calculating the structure of the NLP Jacobian and Hessian.

The above described discretisation scheme is suited for optimal control problems where the derivatives of all the states and controls appear in the equations. When this is not the case, i.e. some states or control are algebraic variables, a slightly modification is required. In order to keep as concise as possible the description of the NLP structure in the following sections, the discussion of the changes required to handle algebraic variables will be postponed to section 6.3.4.

6.3.2 NLP problem structure

In this section the structure of the NLP problem resulting from the optimal control problem (6.5) discretised accordingly to the midpoint method presented in section 6.3.1 is described. As in section 6.3.1, for sake of conciseness a single phase problem is here considered; the generalisation to a multi-phase one is straightforward.

In the previous section the states and controls have been discretised in correspondence of each mesh point. Therefore, all the NLP variables can be grouped into a single vector $\boldsymbol{y}$

as follows:

$$\boldsymbol{y} = \begin{bmatrix} \boldsymbol{y}_0 \\ \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_N \\ \boldsymbol{y}_{N+1} \end{bmatrix}, \qquad \text{where} \qquad \begin{array}{ccc} \boldsymbol{y}_0 & = & [\boldsymbol{x}_0; \boldsymbol{u}_0] \\ \boldsymbol{y}_1 & = & [\boldsymbol{x}_1; \boldsymbol{u}_1] \\ \vdots & & \vdots \\ \boldsymbol{y}_N & = & [\boldsymbol{x}_N; \boldsymbol{u}_N] \\ \boldsymbol{y}_{N+1} & = & \boldsymbol{p} \end{array} \tag{6.17}$$

The number of the NLP optimisation variables n_y (i.e. the size of the vector $\boldsymbol{y}$) is equal to:

$$n_y = (n_x + n_u)(N + 1) + n_p \tag{6.18}$$

where n_x, n_u, n_p are respectively the number of the states $\boldsymbol{x}$, controls $\boldsymbol{u}$ and parameters $\boldsymbol{p}$ of the OCP (6.5). The lower $\boldsymbol{y}_l$ and upper $\boldsymbol{y}_u$ bounds for the NLP variables $\boldsymbol{y}$ can be easily obtained from the OCP bounds (6.5h), (6.5i) and (6.5j):

$$\boldsymbol{y}_l = \begin{bmatrix} [\boldsymbol{x}_l; \boldsymbol{u}_l] \\ [\boldsymbol{x}_l; \boldsymbol{u}_l] \\ \vdots \\ [\boldsymbol{x}_l; \boldsymbol{u}_l] \\ \boldsymbol{p}_l \end{bmatrix}, \qquad \boldsymbol{y}_u = \begin{bmatrix} [\boldsymbol{x}_u; \boldsymbol{u}_u] \\ [\boldsymbol{x}_u; \boldsymbol{u}_u] \\ \vdots \\ [\boldsymbol{x}_u; \boldsymbol{u}_u] \\ \boldsymbol{p}_u \end{bmatrix} \tag{6.19}$$

The NLP target J is given by the sum of the Mayer and Lagrange target, and, accordingly to the midpoint discretisation, it is equal to:

$$J = \psi(\boldsymbol{x}_o, \boldsymbol{x}_N, \boldsymbol{u}_0, \boldsymbol{u}_N, \boldsymbol{p}, \zeta_0, \zeta_N) + \sum_{k=0}^{N-1} \ell_k \theta_k \tag{6.20}$$

where ℓ_k has been defined in (6.14a), and θ_k is the width of the k-th mesh interval.

NLP constraints include the optimal control first order equations (6.14b), path constraints (6.14c), integral constraints (6.14d) and the boundary conditions (6.15b). All the

NLP constraints can be grouped into a single vector $\boldsymbol{c}$ that has the following expression[4]:

$$\boldsymbol{c} = \begin{bmatrix} \boldsymbol{c}_0 \\ \boldsymbol{c}_1 \\ \vdots \\ \boldsymbol{c}_{N-1} \\ \boldsymbol{c}_N \\ \boldsymbol{c}_{N+1} \end{bmatrix} \qquad \text{where} \qquad \begin{array}{ccc} \boldsymbol{c}_0 & = & \left[\boldsymbol{f}_0^\intercal; \boldsymbol{g}_0^\intercal\right] \\ \boldsymbol{c}_1 & = & \left[\boldsymbol{f}_1^\intercal; \boldsymbol{g}_1^\intercal\right] \\ \vdots & & \vdots \\ \boldsymbol{c}_{N-1} & = & \left[\boldsymbol{f}_{N-1}^\intercal; \boldsymbol{g}_{N-1}^\intercal\right] \\ \boldsymbol{c}_N & = & \sum_{k=0}^{N-1} \boldsymbol{h}_k \theta_k \\ \boldsymbol{c}_{N+1} & = & \boldsymbol{\phi}(\boldsymbol{x}_o, \boldsymbol{x}_N, \boldsymbol{u}_0, \boldsymbol{u}_N, \boldsymbol{p}, \zeta_0, \zeta_N) \end{array} \tag{6.21}$$

The number of NLP constraints n_c is equal to:

$$n_c = (n_f + n_p)N + n_i + n_b \tag{6.22}$$

where n_f, n_p, n_i and n_b are respectively the number of first order equations, path constraints, integral constraints and boundary conditions. The upper $\boldsymbol{c}_u$ and lower $\boldsymbol{c}_l$ bounds for the NLP constraints $\boldsymbol{c}$ can be derived from the bounds for the OCP boundary conditions, path and integral constraints:

$$\boldsymbol{c}_l = \begin{bmatrix} [\mathbf{0}; \boldsymbol{g}_l] \\ [\mathbf{0}; \boldsymbol{g}_l] \\ \vdots \\ [\mathbf{0}; \boldsymbol{g}_l] \\ \boldsymbol{h}_l \\ \boldsymbol{\phi}_l \end{bmatrix}, \qquad \boldsymbol{c}_u = \begin{bmatrix} [\mathbf{0}, \boldsymbol{g}_u] \\ [\mathbf{0}, \boldsymbol{g}_u] \\ \vdots \\ [\mathbf{0}, \boldsymbol{g}_u] \\ \boldsymbol{h}_u \\ \boldsymbol{\phi}_u \end{bmatrix} \tag{6.23}$$

where $\mathbf{0}$ is a zero vector of size n_f.

Summarising, the NLP problem originated from the OCP (6.5) through the discretisation scheme presented in section 6.3.1 is the following:

$$\min_{\boldsymbol{y}} J = \psi(\boldsymbol{x}_o, \boldsymbol{x}_N, \boldsymbol{u}_0, \boldsymbol{u}_N, \boldsymbol{p}, \zeta_0, \zeta_N) + \sum_{k=0}^{N-1} \ell_k \theta_k \tag{6.24a}$$

$$\text{subject to:} \tag{6.24b}$$

$$\boldsymbol{c}_l \leq \boldsymbol{c}(\boldsymbol{y}) \leq \boldsymbol{c}_u \tag{6.24c}$$

[4] A different choice could be to multiply the NLP constraints by the mesh interval amplitude θ_k, i.e. $\boldsymbol{c}_k = \theta_k\,[\boldsymbol{f}_k; \boldsymbol{c}_k]$. The motivation for this choice is clear when the equations are explicit $\boldsymbol{x}' = \boldsymbol{f}$, because this results in equating to zero the integration error, i.e. $\delta\boldsymbol{x} = \theta_k \boldsymbol{f}$. The two choices lead to different meaning of the NLP constraints and, thus, of the associated Lagrange multipliers. In section 6.3.3 it will be shown how they affect the relationship between OCP and NLP multipliers.

$$\boldsymbol{y}_l \leq \boldsymbol{y} \leq \boldsymbol{y}_u \tag{6.24d}$$

where J is the NLP target (6.20), $\boldsymbol{y}$ are the NLP variables (6.17) and $\boldsymbol{c}$ are the NLP constraints (6.21).

In order to efficiently solve an NLP problem, not only its target and constraints has to be known, but also the target gradient, the constraints Jacobian and the Lagrangian Hessian should be provided. In the following pages the structure of the NLP Jacobian and Hessian are described.

6.3.2.1 NLP target gradient

The gradient $\nabla_{\boldsymbol{y}} J$ of the target J (6.20) with respect to the NLP variables $\boldsymbol{y}$ (6.17) can be calculated simply by performing the derivatives of the discretised target with respect to the NLP variables. The target gradient thus has the following structure:

$$\nabla_{\boldsymbol{y}} J = \begin{bmatrix} \nabla_{\boldsymbol{x}_0} J \\ \nabla_{\boldsymbol{u}_0} J \\ \nabla_{\boldsymbol{x}_1} J \\ \nabla_{\boldsymbol{u}_1} J \\ \vdots \\ \nabla_{\boldsymbol{x}_{N-1}} J \\ \nabla_{\boldsymbol{u}_{N-1}} J \\ \nabla_{\boldsymbol{x}_N} J \\ \nabla_{\boldsymbol{u}_N} J \\ \nabla_{\boldsymbol{p}} J \end{bmatrix} = \begin{bmatrix} \nabla_{\boldsymbol{x}_0} \psi + \nabla_{\boldsymbol{x}_0} \ell_0 \theta_0 \\ \nabla_{\boldsymbol{u}_0} \psi + \nabla_{\boldsymbol{u}_0} \ell_0 \theta_0 \\ \theta_0 \nabla_{\boldsymbol{x}_1} \ell_0 + \theta_1 \nabla_{\boldsymbol{x}_1} \ell_1 \\ \theta_0 \nabla_{\boldsymbol{u}_1} \ell_0 + \theta_1 \nabla_{\boldsymbol{u}_1} \ell_1 \\ \vdots \\ \theta_{N-2} \nabla_{\boldsymbol{x}_{N-1}} \ell_{N-2} + \theta_{N-1} \nabla_{\boldsymbol{x}_{N-1}} \ell_{N-1} \\ \theta_{N-2} \nabla_{\boldsymbol{u}_{N-1}} \ell_{N-2} + \theta_{N-1} \nabla_{\boldsymbol{u}_{N-1}} \ell_{N-1} \\ \theta_{N-1} \nabla_{\boldsymbol{x}_N} \ell_{N-1} \\ \theta_{N-1} \nabla_{\boldsymbol{u}_N} \ell_{N-1} \\ \nabla_{\boldsymbol{p}} \psi + \sum_{k=0}^{N-1} \theta_k \nabla_{\boldsymbol{p}} \ell_k \end{bmatrix} \tag{6.25}$$

The sparse structure of the target gradient is consequence of the "local" integration scheme adopted: indeed $\nabla_{\boldsymbol{x}_j} \ell_k = 0$ and $\nabla_{\boldsymbol{u}_j} \ell_k = 0$ for all $j \neq k$ and $j \neq k+1$ because ℓ_k depends only on $\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{x}_{k+1}, \boldsymbol{u}_{k+1}$ and $\boldsymbol{p}$.

It can be noted that in the target gradient (6.25), the derivatives are taken with respect to the NLP variables. However it is more convenient to express the derivatives in terms of the optimal control problem variables $\boldsymbol{x}$, $\boldsymbol{x}'$, $\boldsymbol{u}$, $\boldsymbol{u}'$ and $\boldsymbol{p}$. In such way, the target gradient can be easily evaluated once the derivatives of the optimal control problem target (6.5a), (6.5b) are known. Substituting equations (6.16) into (6.25), the NLP target gradient can

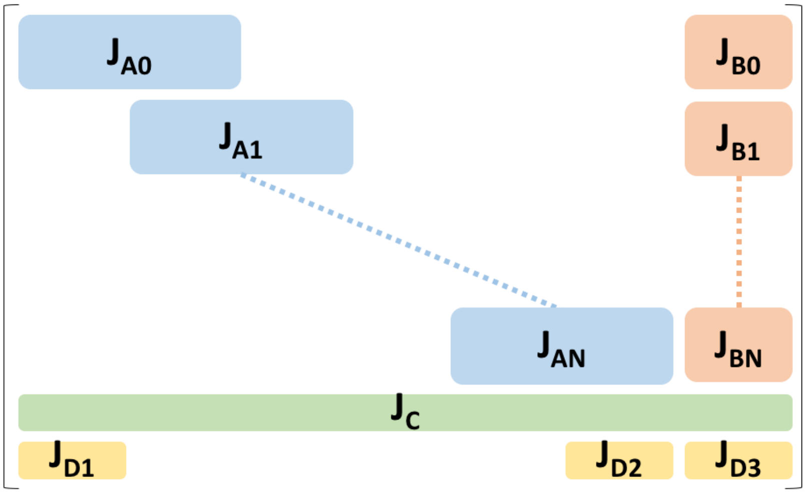


Figure 6.2: Block structure of the NLP constraints Jacobian.

be expressed as follows:

$$\nabla_{\boldsymbol{y}} J = \begin{bmatrix} \nabla_{\boldsymbol{x}_0} J \\ \nabla_{\boldsymbol{u}_0} J \\ \nabla_{\boldsymbol{x}_1} J \\ \nabla_{\boldsymbol{u}_1} J \\ \vdots \\ \nabla_{\boldsymbol{x}_{N-1}} J \\ \nabla_{\boldsymbol{u}_{N-1}} J \\ \nabla_{\boldsymbol{x}_N} J \\ \nabla_{\boldsymbol{u}_N} J \\ \nabla_{\boldsymbol{p}} J \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\left(\frac{\partial \psi}{\partial \boldsymbol{x}_i} + \frac{\partial \ell_0}{\partial \boldsymbol{x}}\right) - \frac{1}{\theta_0}\frac{\partial \ell_0}{\partial \boldsymbol{x}'} \\ \frac{1}{2}\left(\frac{\partial \psi}{\partial \boldsymbol{u}_i} + \frac{\partial \ell_0}{\partial \boldsymbol{u}}\right) - \frac{1}{\theta_0}\frac{\partial \ell_0}{\partial \boldsymbol{u}'} \\ \frac{1}{2}\left(\frac{\partial \ell_0}{\partial \boldsymbol{x}} + \frac{\partial \ell_1}{\partial \boldsymbol{x}}\right) - \frac{1}{\theta_1}\frac{\partial \ell_1}{\partial \boldsymbol{x}'} + \frac{1}{\theta_0}\frac{\partial \ell_0}{\partial \boldsymbol{x}'} \\ \frac{1}{2}\left(\frac{\partial \ell_0}{\partial \boldsymbol{u}} + \frac{\partial \ell_1}{\partial \boldsymbol{u}}\right) - \frac{1}{\theta_1}\frac{\partial \ell_1}{\partial \boldsymbol{u}'} + \frac{1}{\theta_0}\frac{\partial \ell_0}{\partial \boldsymbol{u}'} \\ \vdots \\ \frac{\theta_{N-2}}{2}\frac{\partial \ell_{N-2}}{\partial \boldsymbol{x}} + \frac{\theta_{N-1}}{2}\frac{\partial \ell_{N-1}}{\partial \boldsymbol{x}} - \frac{\partial \ell_{N-2}}{\partial \boldsymbol{x}'} + \frac{\partial \ell_{N-1}}{\partial \boldsymbol{x}'} \\ \frac{\theta_{N-2}}{2}\frac{\partial \ell_{N-2}}{\partial \boldsymbol{u}} + \frac{\theta_{N-1}}{2}\frac{\partial \ell_{N}}{\partial \boldsymbol{u}} - \frac{\partial \ell_{N-2}}{\partial \boldsymbol{u}'} + \frac{\partial \ell_{N-1}}{\partial \boldsymbol{u}'} \\ \frac{1}{2}\left(\frac{\partial \psi}{\partial \boldsymbol{x}_f} + \theta_{N-1}\frac{\partial \ell_{N-1}}{\partial \boldsymbol{x}}\right) + \frac{\partial \ell_{N-1}}{\partial \boldsymbol{x}'} \\ \frac{1}{2}\left(\frac{\partial \psi}{\partial \boldsymbol{u}_f} + \theta_{N-1}\frac{\partial \ell_{N-1}}{\partial \boldsymbol{u}}\right) + \frac{\partial \ell_{N-1}}{\partial \boldsymbol{u}'} \\ \frac{\partial \psi}{\partial \boldsymbol{p}} + \sum_{k=0}^{N-1} \theta_k \frac{\partial \ell_k}{\partial \boldsymbol{p}} \end{bmatrix} \tag{6.26}$$

6.3.2.2 NLP constraint Jacobian

The NLP constraint Jacobian matrix has n_c rows and n_y columns, and in real optimal control problems its size can be very large. However, thanks to the local structure of the midpoint discretisation scheme, the Jacobian matrix is sparse and most of its entries are identically zero.

NLP constraints include first order equations $\boldsymbol{f}_k$, path constraints $\boldsymbol{g}_k$, integral constraints $\boldsymbol{h}$ and the boundary conditions $\boldsymbol{\phi}$. Accordingly to equation (6.16), $\boldsymbol{f}_k$, $\boldsymbol{g}_k$, and $\boldsymbol{h}$ depend on few NLP variables ($\boldsymbol{x}_k$, $\boldsymbol{x}_{k+1}$, $\boldsymbol{u}_k$, $\boldsymbol{u}_{k+1}$); moreover also the boundary conditions depend on few variables:

$$\frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{x}_j} = 0 \qquad \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{u}_j} = 0 \qquad \text{for } j \neq 0 \text{ and } j \neq N \tag{6.27}$$

The resulting NLP constraint Jacobian matrix $\boldsymbol{J}$ thus has the structure shown in figure 6.2, where only the diagonal part and the right and bottom bands are filled. Matrix blocks named $\boldsymbol{J}_{Ak}$ and $\boldsymbol{J}_{Bk}$ with $k = 0, \dots, N$ are originated from the first order equations and path constraints, the bottom block $\boldsymbol{J}_C$ is the Jacobian of the integral constraints and the blocks $\boldsymbol{J}_{D1}$, $\boldsymbol{J}_{D2}$, $\boldsymbol{J}_{D3}$ are related to the boundary conditions.

Each block $\boldsymbol{J}_{Ak}$ is the Jacobian of the k-th first order equations $\boldsymbol{f}_k$ and path constraints $\boldsymbol{g}_k$ with respect to the discrete variables $\boldsymbol{x}_k$, $\boldsymbol{u}_k$, $\boldsymbol{x}_{k+1}$ and $\boldsymbol{u}_{k+1}$, while $\boldsymbol{J}_{Bk}$ is the Jacobian with respect to the optimisation parameters $\boldsymbol{p}$:

$$\begin{aligned} \boldsymbol{J}_{Ak} &= \begin{bmatrix} \frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{x}_k} & \frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{u}_k} & \frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{x}_{k+1}} & \frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{u}_{k+1}} & \frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{x}_k} & \frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{u}_k} & \frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{x}_{k+1}} & \frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{u}_{k+1}} \end{bmatrix} \\ \boldsymbol{J}_{Bk} &= \begin{bmatrix} \frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{p}} \\ \frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{p}} \end{bmatrix} \end{aligned} \tag{6.28}$$

As in the case of the gradient target (6.25), it is more convenient to express the Jacobian in terms of derivatives taken with respect to the the optimal control problem variables $\boldsymbol{x}$, $\boldsymbol{x}'$, $\boldsymbol{u}$, $\boldsymbol{u}'$ and $\boldsymbol{p}$. Substituting (6.16) into (6.28), the block $\boldsymbol{J}_{Ak}$ becomes:

$$\boldsymbol{J}_{Ak} = \begin{bmatrix} \frac{1}{2}\frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{x}} - \frac{1}{\theta_k}\frac{\partial \boldsymbol{f}_k}{\partial \dot{\boldsymbol{x}}} & \frac{1}{2}\frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{u}} - \frac{1}{\theta_k}\frac{\partial \boldsymbol{f}_k}{\partial \dot{\boldsymbol{u}}} & \frac{1}{2}\frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{x}} + \frac{1}{\theta_k}\frac{\partial \boldsymbol{f}_k}{\partial \dot{\boldsymbol{x}}} & \frac{1}{2}\frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{u}} + \frac{1}{\theta_k}\frac{\partial \boldsymbol{f}_k}{\partial \dot{\boldsymbol{u}}} \\ \frac{1}{2}\frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{x}} - \frac{1}{\theta_k}\frac{\partial \boldsymbol{g}_k}{\partial \dot{\boldsymbol{x}}} & \frac{1}{2}\frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{u}} - \frac{1}{\theta_k}\frac{\partial \boldsymbol{g}_k}{\partial \dot{\boldsymbol{u}}} & \frac{1}{2}\frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{x}} + \frac{1}{\theta_k}\frac{\partial \boldsymbol{g}_k}{\partial \dot{\boldsymbol{x}}} & \frac{1}{2}\frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{u}} + \frac{1}{\theta_k}\frac{\partial \boldsymbol{g}_k}{\partial \dot{\boldsymbol{u}}} \end{bmatrix} \tag{6.29}$$

The Jacobian of the integral constraints $\boldsymbol{J}_C$ has instead the following structure:

$$
\begin{aligned}
\boldsymbol{J}_C &= \begin{bmatrix} \frac{\partial \boldsymbol{h}_0}{\partial \boldsymbol{x}_0} & \frac{\partial \boldsymbol{h}_0}{\partial \boldsymbol{u}_0} & \frac{\partial \boldsymbol{h}_0+\boldsymbol{h}_1}{\partial \boldsymbol{x}_1} & \frac{\partial \boldsymbol{h}_0+\boldsymbol{h}_1}{\partial \boldsymbol{u}_1} & \cdots & \frac{\partial \boldsymbol{h}_{N-1}+\boldsymbol{h}_N}{\partial \boldsymbol{x}_{N-1}} & \frac{\partial \boldsymbol{h}_{N-1}+\boldsymbol{h}_N}{\partial \boldsymbol{u}_{N-1}} & \frac{\partial \boldsymbol{h}_N}{\partial \boldsymbol{x}_N} & \frac{\partial \boldsymbol{h}_N}{\partial \boldsymbol{u}_N} \end{bmatrix} \\
&= \begin{bmatrix}
\frac{\theta_0}{2}\frac{\partial \boldsymbol{h}_0}{\partial \boldsymbol{x}} - \frac{\partial \boldsymbol{h}_0}{\partial \boldsymbol{x}'} \\
\frac{\theta_0}{2}\frac{\partial \boldsymbol{h}_0}{\partial \boldsymbol{u}} - \frac{\partial \boldsymbol{h}_0}{\partial \boldsymbol{u}'} \\
\frac{\theta_0}{2}\frac{\partial \boldsymbol{h}_0}{\partial \boldsymbol{x}} + \frac{\theta_1}{2}\frac{\partial \boldsymbol{h}_1}{\partial \boldsymbol{x}} + \left(\frac{\partial \boldsymbol{h}_0}{\partial \boldsymbol{x}'} - \frac{\partial \boldsymbol{h}_1}{\partial \boldsymbol{x}'}\right) \\
\frac{\theta_0}{2}\frac{\partial \boldsymbol{h}_0}{\partial \boldsymbol{u}} + \frac{\theta_1}{2}\frac{\partial \boldsymbol{h}_1}{\partial \boldsymbol{u}} + \left(\frac{\partial \boldsymbol{h}_0}{\partial \boldsymbol{u}'} - \frac{\partial \boldsymbol{h}_1}{\partial \boldsymbol{u}'}\right) \\
\vdots \\
\frac{\theta_{N-2}}{2}\frac{\partial \boldsymbol{h}_{N-2}}{\partial \boldsymbol{x}} + \frac{\theta_{N-1}}{2}\frac{\partial \boldsymbol{h}_{N-1}}{\partial \boldsymbol{x}} + \left(\frac{\partial \boldsymbol{h}_{N-2}}{\partial \boldsymbol{x}'} - \frac{\partial \boldsymbol{h}_{N-1}}{\partial \boldsymbol{x}'}\right) \\
\frac{\theta_{N-2}}{2}\frac{\partial \boldsymbol{h}_{N-2}}{\partial \boldsymbol{u}} + \frac{\theta_{N-1}}{2}\frac{\partial \boldsymbol{h}_{N-1}}{\partial \boldsymbol{u}} + \left(\frac{\partial \boldsymbol{h}_{N-2}}{\partial \boldsymbol{u}'} - \frac{\partial \boldsymbol{h}_{N-1}}{\partial \boldsymbol{u}'}\right) \\
\frac{\theta_{N-1}}{2}\frac{\partial \boldsymbol{h}_{N-1}}{\partial \boldsymbol{x}} - \frac{\partial \boldsymbol{h}_{N-1}}{\partial \boldsymbol{x}'} \\
\frac{\theta_{N-1}}{2}\frac{\partial \boldsymbol{h}_{N-1}}{\partial \boldsymbol{u}} - \frac{\partial \boldsymbol{h}_{N-1}}{\partial \boldsymbol{u}'}
\end{bmatrix}^{\mathsf{T}}
\end{aligned}
\tag{6.30}
$$

Finally, blocks $\boldsymbol{J}_{D1}$, $\boldsymbol{J}_{D2}$ and $\boldsymbol{J}_{D3}$, that are related to the boundary conditions, are equal to:

$$
\begin{aligned}
\boldsymbol{J}_{D1} &= \begin{bmatrix} \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{x}_0} & \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{u}_0} \end{bmatrix} &&= \begin{bmatrix} \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{x}_i} & \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{u}_i} \end{bmatrix} \\
\boldsymbol{J}_{D2} &= \begin{bmatrix} \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{x}_N} & \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{u}_N} \end{bmatrix} &&= \begin{bmatrix} \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{x}_f} & \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{u}_f} \end{bmatrix} \\
\boldsymbol{J}_{D3} &= \begin{bmatrix} \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{p}} \end{bmatrix}
\end{aligned}
\tag{6.31}
$$

6.3.2.3 NLP Lagrangian Hessian

The NLP Lagrangian function $\mathcal{L}$ is defined as the sum of the NLP target and of the NLP constraints, each multiplied by its corresponding (NLP) Lagrange multiplier:

$$
\mathcal{L} = J + \sum_{k=0}^{N-1} \left(\boldsymbol{\Lambda}_k^{\mathsf{T}} \boldsymbol{f}_k + \boldsymbol{\Omega}_k^{\mathsf{T}} \boldsymbol{g}_k \right) + \boldsymbol{\Upsilon}^{\mathsf{T}} \sum_{k=0}^{N-1} \left(\boldsymbol{h}_k \theta_k \right) + \boldsymbol{\gamma}^{\mathsf{T}} \boldsymbol{\phi} \tag{6.32}
$$

where J is the NLP target (6.20), $\boldsymbol{\Lambda}_k, \boldsymbol{\Omega}_k, \boldsymbol{\Upsilon}$ and $\boldsymbol{\gamma}$ are the Lagrange multipliers associated respectively to the first order equations $\boldsymbol{f}_k$, path constraints $\boldsymbol{g}_k$, integral constraints $\boldsymbol{h}$ and boundary conditions $\boldsymbol{\phi}$[5].

The solution of an NLP problem can be calculated in less time and more robustly if the Hessian of the NLP Lagrangian function $\mathcal{L}$ (6.32) with respect to the NLP variables $\boldsymbol{y}$ (6.17) is known. Numerical NLP solvers indeed make use of the NLP Lagrangian Hessian when the user can provide it. The structure of the Lagrangian Hessian matrix for the NLP

[5] Previously bold lower-case greek symbols have been used for the OCP Lagrange multipliers, here the corresponding upper-case greek symbols are used for the NLP Lagrange multipliers. In this case bold upper-case symbols denote vectors, not matrices.

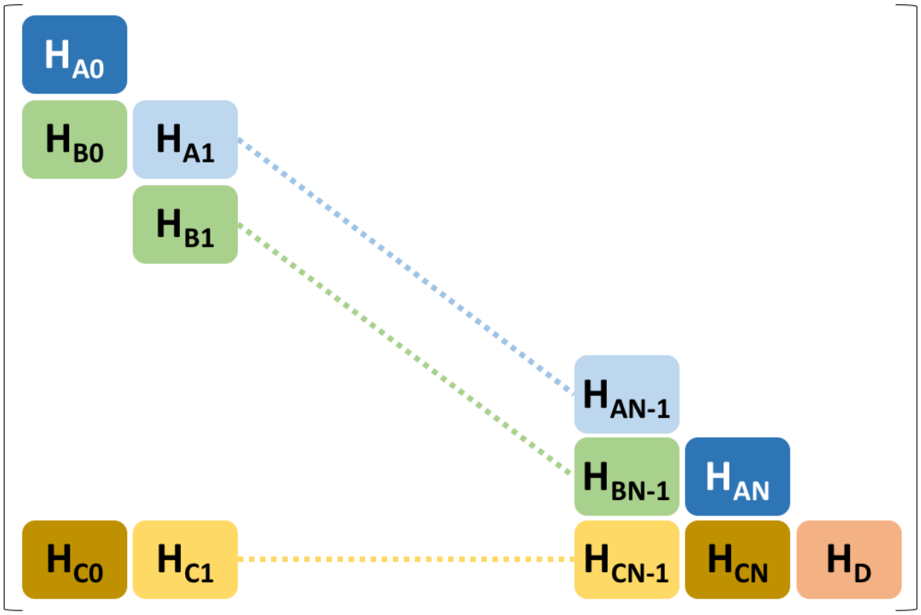


Figure 6.3: Block structure of the NLP Lagrangian Hessian. The upper triangular part is not shown because the matrix is symmetric.

problem (6.24) is now described.

The Lagrangian Hessian is a square symmetric matrix of dimensions $n_y \times n_y$ and, as the NLP constraints Jacobian, it is sparse and most of its entries are identically zero (thanks to the structure of the NLP obtained through the discretisation scheme). The block structure of the Hessian matrix is shown in figure 6.3; it can be noted that, similarly to the Jacobian matrix, only the part close to its diagonal and a lower band [6] are filled. With reference to figure 6.3, matrix blocks named $\boldsymbol{H}_{Ak}$ and $\boldsymbol{H}_{Bk}$ are related to the second derivatives of the NLP target and NLP constraints $\boldsymbol{g}_k$, $\boldsymbol{g}_{k+1}$, $\boldsymbol{g}_N$ (that include first order equations, path and integral constraints):

$$\begin{aligned}
\boldsymbol{H}_{Ak} &= \theta_k \frac{\partial^2 \left(\ell_k + \boldsymbol{\Upsilon}^\intercal \boldsymbol{h}_k\right)}{\partial \boldsymbol{v}_k^2} + \theta_{k-1} \frac{\partial^2 \left(\ell_{k-1} + \boldsymbol{\Upsilon}^\intercal \boldsymbol{h}_{k-1}\right)}{\partial \boldsymbol{v}_k^2} \\
&\quad + \frac{\partial^2 \left(\boldsymbol{\Lambda}_k^\intercal \boldsymbol{f}_k + \boldsymbol{\Omega}_k^\intercal \boldsymbol{g}_k\right)}{\partial \boldsymbol{v}_k^2} + \frac{\partial^2 \left(\boldsymbol{\Lambda}_{k-1}^\intercal \boldsymbol{f}_{k-1} + \boldsymbol{\Omega}_{k-1}^\intercal \boldsymbol{g}_{k-1}\right)}{\partial \boldsymbol{v}_k^2} \qquad \text{for } k = 1, \dots, N-1 \\
\boldsymbol{H}_{Bk} &= \theta_k \frac{\partial^2 \left(\ell_k + \boldsymbol{\Upsilon}^\intercal \boldsymbol{h}_k\right)}{\partial \boldsymbol{v}_{k+1} \partial \boldsymbol{v}_k^\intercal} + \frac{\partial^2 \left(\boldsymbol{\Lambda}_k^\intercal \boldsymbol{f}_k + \boldsymbol{\Omega}_k^\intercal \boldsymbol{g}_k\right)}{\partial \boldsymbol{v}_{k+1} \partial \boldsymbol{v}_k^\intercal} \qquad \text{for } k = 0, \dots, N-1
\end{aligned} \tag{6.33}$$

where $\boldsymbol{v}_k$ is a vector (used here for a compact notation) that includes both the discretised

[6] Since the Hessian is symmetric, there is also a filled band in the right-most part of the matrix, which is equal to the transpose of the lower band.

states $\boldsymbol{x}_k$ and controls $\boldsymbol{u}_k$: $\boldsymbol{v}_k = [\boldsymbol{x}_k; \boldsymbol{u}_k]$. The first $\boldsymbol{H}_{A0}$ and last $\boldsymbol{H}_{AN}$ blocks have a different expression due to the presence of the boundary conditions and Mayer target:

$$\begin{aligned}
\boldsymbol{H}_{A0} &= \theta_0 \frac{\partial^2 \left(\ell_0 + \boldsymbol{\Upsilon}^\mathsf{T} \boldsymbol{h}_0\right)}{\partial \boldsymbol{v}_0^2} + \frac{\partial^2 \left(\boldsymbol{\Lambda}_0^\mathsf{T} \boldsymbol{f}_0 + \boldsymbol{\Omega}_0^\mathsf{T} \boldsymbol{g}_0 + \psi + \boldsymbol{\gamma}^\mathsf{T} \boldsymbol{\phi}\right)}{\partial \boldsymbol{v}_0^2} \\
\boldsymbol{H}_{AN} &= \theta_N \frac{\partial^2 \left(\ell_N + \boldsymbol{\Upsilon}^\mathsf{T} \boldsymbol{h}_N\right)}{\partial \boldsymbol{v}_N^2} + \frac{\partial^2 \left(\boldsymbol{\Lambda}_N^\mathsf{T} \boldsymbol{f}_N + \boldsymbol{\Omega}_N^\mathsf{T} \boldsymbol{g}_N + \psi + \boldsymbol{\gamma}^\mathsf{T} \boldsymbol{\phi}\right)}{\partial \boldsymbol{v}_N^2}
\end{aligned} \tag{6.34}$$

Each block $\boldsymbol{H}_{Ak}$ and $\boldsymbol{H}_{Bk}$ can be written in a more explicit form by substituting twice (6.16) in (6.33) and (6.34):

$$\begin{aligned}
\boldsymbol{H}_{Ak} = {} & \frac{\theta_k}{4} \frac{\partial^2 \left(\ell_k + \boldsymbol{\Upsilon}^\mathsf{T} \boldsymbol{h}_k\right)}{\partial \boldsymbol{v}^2} + \frac{1}{\theta_k} \frac{\partial^2 \left(\ell_k + \boldsymbol{\Upsilon}^\mathsf{T} \boldsymbol{h}_k\right)}{\partial \boldsymbol{v}'^2} - \frac{1}{2} \frac{\partial^2 \left(\ell_k + \boldsymbol{\Upsilon}^\mathsf{T} \boldsymbol{h}_k\right)}{\partial \boldsymbol{v} \partial \boldsymbol{v}'^\mathsf{T}} - \frac{1}{2} \frac{\partial^2 \left(\ell_k + \boldsymbol{\Upsilon}^\mathsf{T} \boldsymbol{h}_k\right)}{\partial \boldsymbol{v}' \partial \boldsymbol{v}^\mathsf{T}} \\
& + \frac{\theta_{k-1}}{4} \frac{\partial^2 \left(\ell_{k-1} + \boldsymbol{\Upsilon}^\mathsf{T} \boldsymbol{h}_{k-1}\right)}{\partial \boldsymbol{v}^2} + \frac{1}{\theta_{k-1}} \frac{\partial^2 \left(\ell_{k-1} + \boldsymbol{\Upsilon}^\mathsf{T} \boldsymbol{h}_{k-1}\right)}{\partial \boldsymbol{v}'^2} \\
& + \frac{1}{2} \frac{\partial^2 \left(\ell_{k-1} + \boldsymbol{\Upsilon}^\mathsf{T} \boldsymbol{h}_{k-1}\right)}{\partial \boldsymbol{v} \partial \boldsymbol{v}'^\mathsf{T}} + \frac{1}{2} \frac{\partial^2 \left(\ell_{k-1} + \boldsymbol{\Upsilon}^\mathsf{T} \boldsymbol{h}_{k-1}\right)}{\partial \boldsymbol{v}' \partial \boldsymbol{v}^\mathsf{T}} \\
& + \frac{1}{4} \frac{\partial^2 \left(\boldsymbol{\Lambda}_k^\mathsf{T} \boldsymbol{f}_k + \boldsymbol{\Omega}_k^\mathsf{T} \boldsymbol{g}_k\right)}{\partial \boldsymbol{v}^2} + \frac{1}{\theta_k^2} \frac{\partial^2 \left(\boldsymbol{\Lambda}_k^\mathsf{T} \boldsymbol{f}_k + \boldsymbol{\Omega}_k^\mathsf{T} \boldsymbol{g}_k\right)}{\partial \boldsymbol{v}'^2} \\
& - \frac{1}{2\theta_k} \frac{\partial^2 \left(\boldsymbol{\Lambda}_k^\mathsf{T} \boldsymbol{f}_k + \boldsymbol{\Omega}_k^\mathsf{T} \boldsymbol{g}_k\right)}{\partial \boldsymbol{v} \partial \boldsymbol{v}'^\mathsf{T}} - \frac{1}{2\theta_k} \frac{\partial^2 \left(\boldsymbol{\Lambda}_k^\mathsf{T} \boldsymbol{f}_k + \boldsymbol{\Omega}_k^\mathsf{T} \boldsymbol{g}_k\right)}{\partial \boldsymbol{v}' \partial \boldsymbol{v}^\mathsf{T}} \\
& + \frac{1}{4} \frac{\partial^2 \left(\boldsymbol{\Lambda}_{k-1}^\mathsf{T} \boldsymbol{f}_{k-1} + \boldsymbol{\Omega}_{k-1}^\mathsf{T} \boldsymbol{g}_{k-1}\right)}{\partial \boldsymbol{v}^2} + \frac{1}{\theta_{k-1}^2} \frac{\partial^2 \left(\boldsymbol{\Lambda}_{k-1}^\mathsf{T} \boldsymbol{f}_{k-1} + \boldsymbol{\Omega}_{k-1}^\mathsf{T} \boldsymbol{g}_{k-1}\right)}{\partial \boldsymbol{v}'^2} \\
& + \frac{1}{2\theta_{k-1}} \frac{\partial^2 \left(\boldsymbol{\Lambda}_{k-1}^\mathsf{T} \boldsymbol{f}_{k-1} + \boldsymbol{\Omega}_{k-1}^\mathsf{T} \boldsymbol{g}_{k-1}\right)}{\partial \boldsymbol{v} \partial \boldsymbol{v}'^\mathsf{T}} + \frac{1}{2\theta_{k-1}} \frac{\partial^2 \left(\boldsymbol{\Lambda}_{k-1}^\mathsf{T} \boldsymbol{f}_{k-1} + \boldsymbol{\Omega}_{k-1}^\mathsf{T} \boldsymbol{g}_{k-1}\right)}{\partial \boldsymbol{v}' \partial \boldsymbol{v}^\mathsf{T}}
\end{aligned} \tag{6.35a}$$

$$\begin{aligned}
\boldsymbol{H}_{Bk} = {} & \frac{\theta_k}{4} \frac{\partial^2 \left(\ell_k + \boldsymbol{\Upsilon}^\mathsf{T} \boldsymbol{h}_k\right)}{\partial \boldsymbol{v}^2} - \frac{1}{\theta_k} \frac{\partial^2 \left(\ell_k + \boldsymbol{\Upsilon}^\mathsf{T} \boldsymbol{h}_k\right)}{\partial \boldsymbol{v}'^2} + \frac{1}{2} \frac{\partial^2 \left(\ell_k + \boldsymbol{\Upsilon}^\mathsf{T} \boldsymbol{h}_k\right)}{\partial \boldsymbol{v} \partial \boldsymbol{v}'^\mathsf{T}} - \frac{1}{2} \frac{\partial^2 \left(\ell_k + \boldsymbol{\Upsilon}^\mathsf{T} \boldsymbol{h}_k\right)}{\partial \boldsymbol{v}' \partial \boldsymbol{v}^\mathsf{T}} \\
& + \frac{1}{4} \frac{\partial^2 \left(\boldsymbol{\Lambda}_k^\mathsf{T} \boldsymbol{f}_k + \boldsymbol{\Omega}_k^\mathsf{T} \boldsymbol{g}_k\right)}{\partial \boldsymbol{v}^2} - \frac{1}{\theta_k^2} \frac{\partial^2 \left(\boldsymbol{\Lambda}_k^\mathsf{T} \boldsymbol{f}_k + \boldsymbol{\Omega}_k^\mathsf{T} \boldsymbol{g}_k\right)}{\partial \boldsymbol{v}'^2} \\
& + \frac{1}{2\theta_k} \frac{\partial^2 \left(\boldsymbol{\Lambda}_k^\mathsf{T} \boldsymbol{f}_k + \boldsymbol{\Omega}_k^\mathsf{T} \boldsymbol{g}_k\right)}{\partial \boldsymbol{v} \partial \boldsymbol{v}'^\mathsf{T}} - \frac{1}{2\theta_k} \frac{\partial^2 \left(\boldsymbol{\Lambda}_k^\mathsf{T} \boldsymbol{f}_k + \boldsymbol{\Omega}_k^\mathsf{T} \boldsymbol{g}_k\right)}{\partial \boldsymbol{v}' \partial \boldsymbol{v}^\mathsf{T}}
\end{aligned} \tag{6.35b}$$

$$
\begin{aligned}
\boldsymbol{H}_{A0} = {} & \frac{\theta_0}{4}\frac{\partial^2\left(\ell_0 + \boldsymbol{\Upsilon}^{\mathsf{T}}\boldsymbol{h}_0\right)}{\partial \boldsymbol{v}^2} + \frac{1}{4}\frac{\partial^2\left(\psi + \boldsymbol{\gamma}^{\mathsf{T}}\boldsymbol{\phi}\right)}{\partial \boldsymbol{v}_i^2} + \frac{1}{\theta_0}\frac{\partial^2\left(\ell_0 + \boldsymbol{\Upsilon}^{\mathsf{T}}\boldsymbol{h}_0\right)}{\partial \boldsymbol{v}'^2} + \frac{1}{\theta_0^2}\frac{\partial^2\left(\psi + \boldsymbol{\gamma}^{\mathsf{T}}\boldsymbol{\phi}\right)}{\partial \boldsymbol{v}_i'^2} \\
& - \frac{1}{2}\frac{\partial^2\left(\ell_0 + \boldsymbol{\Upsilon}^{\mathsf{T}}\boldsymbol{h}_0\right)}{\partial \boldsymbol{v}\partial \boldsymbol{v}'^{\mathsf{T}}} - \frac{1}{2\theta_0}\frac{\partial^2\left(\psi + \boldsymbol{\gamma}^{\mathsf{T}}\boldsymbol{\phi}\right)}{\partial \boldsymbol{v}_i\partial \boldsymbol{v}_i'^{\mathsf{T}}} - \frac{1}{2}\frac{\partial^2\left(\ell_0 + \boldsymbol{\Upsilon}^{\mathsf{T}}\boldsymbol{h}_0\right)}{\partial \boldsymbol{v}'\partial \boldsymbol{v}^{\mathsf{T}}} - \frac{1}{2\theta_0}\frac{\partial^2\left(\psi + \boldsymbol{\gamma}^{\mathsf{T}}\boldsymbol{\phi}\right)}{\partial \boldsymbol{v}_i'\partial \boldsymbol{v}_i^{\mathsf{T}}} \\
& + \frac{1}{4}\frac{\partial^2\left(\boldsymbol{\Lambda}_0^{\mathsf{T}}\boldsymbol{f}_0 + \boldsymbol{\Omega}_0^{\mathsf{T}}\boldsymbol{g}_0\right)}{\partial \boldsymbol{v}^2} + \frac{1}{\theta_0^2}\frac{\partial^2\left(\boldsymbol{\Lambda}_k^{\mathsf{T}}\boldsymbol{f}_0 + \boldsymbol{\Omega}_0^{\mathsf{T}}\boldsymbol{g}_0\right)}{\partial \boldsymbol{v}'^2} \\
& - \frac{1}{2\theta_0}\frac{\partial^2\left(\boldsymbol{\Lambda}_k^{\mathsf{T}}\boldsymbol{f}_0 + \boldsymbol{\Omega}_0^{\mathsf{T}}\boldsymbol{g}_0\right)}{\partial \boldsymbol{v}\partial \boldsymbol{v}'^{\mathsf{T}}} - \frac{1}{2\theta_0}\frac{\partial^2\left(\boldsymbol{\Lambda}_k^{\mathsf{T}}\boldsymbol{f}_0 + \boldsymbol{\Omega}_0^{\mathsf{T}}\boldsymbol{g}_0\right)}{\partial \boldsymbol{v}'\partial \boldsymbol{v}^{\mathsf{T}}}
\end{aligned}
\tag{6.35c}
$$

$$
\begin{aligned}
\boldsymbol{H}_{AN} = {} & \frac{\theta_{N-1}}{4}\frac{\partial^2\left(\ell_{N-1} + \boldsymbol{\Upsilon}^{\mathsf{T}}\boldsymbol{h}_{N-1}\right)}{\partial \boldsymbol{v}^2} + \frac{1}{4}\frac{\partial^2\left(\psi + \boldsymbol{\gamma}^{\mathsf{T}}\boldsymbol{\phi}\right)}{\partial \boldsymbol{v}_f^2} \\
& + \frac{1}{\theta_{N-1}}\frac{\partial^2\left(\ell_{N-1} + \boldsymbol{\Upsilon}^{\mathsf{T}}\boldsymbol{h}_{N-1}\right)}{\partial \boldsymbol{v}'^2} + \frac{1}{\theta_{N-1}^2}\frac{\partial^2\left(\psi + \boldsymbol{\gamma}^{\mathsf{T}}\boldsymbol{\phi}\right)}{\partial \boldsymbol{v}_f'^2} \\
& + \frac{1}{2}\frac{\partial^2\left(\ell_{N-1} + \boldsymbol{\Upsilon}^{\mathsf{T}}\boldsymbol{h}_{N-1}\right)}{\partial \boldsymbol{v}\partial \boldsymbol{v}'^{\mathsf{T}}} + \frac{1}{2\theta_{N-1}}\frac{\partial^2\left(\psi + \boldsymbol{\gamma}^{\mathsf{T}}\boldsymbol{\phi}\right)}{\partial \boldsymbol{v}_f\partial \boldsymbol{v}_f'^{\mathsf{T}}} \\
& + \frac{1}{2}\frac{\partial^2\left(\ell_{N-1} + \boldsymbol{\Upsilon}^{\mathsf{T}}\boldsymbol{h}_{N-1}\right)}{\partial \boldsymbol{v}'\partial \boldsymbol{v}^{\mathsf{T}}} + \frac{1}{2\theta_{N-1}}\frac{\partial^2\left(\psi + \boldsymbol{\gamma}^{\mathsf{T}}\boldsymbol{\phi}\right)}{\partial \boldsymbol{v}_f'\partial \boldsymbol{v}_f^{\mathsf{T}}} \\
& + \frac{1}{4}\frac{\partial^2\left(\boldsymbol{\Lambda}_{N-1}^{\mathsf{T}}\boldsymbol{f}_{N-1} + \boldsymbol{\Omega}_{N-1}^{\mathsf{T}}\boldsymbol{g}_{N-1}\right)}{\partial \boldsymbol{v}^2} + \frac{1}{\theta_k^2}\frac{\partial^2\left(\boldsymbol{\Lambda}_{N-1}^{\mathsf{T}}\boldsymbol{f}_{N-1} + \boldsymbol{\Omega}_{N-1}^{\mathsf{T}}\boldsymbol{g}_{N-1}\right)}{\partial \boldsymbol{v}'^2} \\
& + \frac{1}{2\theta_k}\frac{\partial^2\left(\boldsymbol{\Lambda}_{N-1}^{\mathsf{T}}\boldsymbol{f}_{N-1} + \boldsymbol{\Omega}_{N-1}^{\mathsf{T}}\boldsymbol{g}_{N-1}\right)}{\partial \boldsymbol{v}\partial \boldsymbol{v}'^{\mathsf{T}}} + \frac{1}{2\theta_k}\frac{\partial^2\left(\boldsymbol{\Lambda}_{N-1}^{\mathsf{T}}\boldsymbol{f}_{N-1} + \boldsymbol{\Omega}_{N-1}^{\mathsf{T}}\boldsymbol{g}_{N-1}\right)}{\partial \boldsymbol{v}'\partial \boldsymbol{v}^{\mathsf{T}}}
\end{aligned}
\tag{6.35d}
$$

where $\boldsymbol{v}$ is a vector (used here for a compact notation) that includes both the states $\boldsymbol{x}$ and controls $\boldsymbol{u}$: $\boldsymbol{v} = [\boldsymbol{x}; \boldsymbol{u}]$. In (6.35) it can be noted that only the blocks $\boldsymbol{H}_{A0}$ and $\boldsymbol{H}_{AN}$ include the second order derivatives of the boundary conditions $\boldsymbol{g}_{N+1}$ with respect to $\boldsymbol{y}_0$ (i.e. $\boldsymbol{x}_0, \boldsymbol{u}_0$) and $\boldsymbol{y}_N$ (i.e. $\boldsymbol{x}_N, \boldsymbol{u}_N$).

Block in the lower band $\boldsymbol{H}_{C0}, \ldots, \boldsymbol{H}_{CN}$ are related to the second derivatives with respect to optimisation parameters $\boldsymbol{y}_{N+1}$ and the other NLP variables $\boldsymbol{y}_0, \ldots, \boldsymbol{y}_N$.

Hessian blocks $\boldsymbol{H}_{C0}, \ldots, \boldsymbol{H}_{CN}$ in the bottom band of the NLP Hessian (see figure 6.3) are related to the second order derivatives with respect to the NLP parameters $\boldsymbol{y}_{N+1}$ and the other NLP variables $\boldsymbol{y}_0, \ldots, \boldsymbol{y}_N$ In particular, blocks $\boldsymbol{H}_{Ck}$ are very similar to $\boldsymbol{H}_{Ak}$, where the partial derivative $\partial \boldsymbol{v}_k$ is replaced by $\partial \boldsymbol{p}$:

$$
\begin{aligned}
\boldsymbol{H}_{Ck} &= \theta_k \frac{\partial^2 \left(\ell_k + \boldsymbol{\Upsilon}^\intercal \boldsymbol{h}_k\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}_k 1tr} + \theta_{k-1} \frac{\partial^2 \left(\ell_{k-1} + \boldsymbol{\Upsilon}^\intercal \boldsymbol{h}_{k-1}\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}_k^\intercal} \\
&\quad + \frac{\partial^2 \left(\boldsymbol{\Lambda}_k^\intercal \boldsymbol{f}_k + \boldsymbol{\Omega}_k^\intercal \boldsymbol{g}_k\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}_k^\intercal} + \frac{\partial^2 \left(\boldsymbol{\Lambda}_{k-1}^\intercal \boldsymbol{f}_{k-1} + \boldsymbol{\Omega}_{k-1}^\intercal \boldsymbol{g}_{k-1}\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}_k^\intercal} \qquad \text{for } k = 1, \ldots, N-1 \\
\boldsymbol{H}_{C0} &= \theta_0 \frac{\partial^2 \left(\ell_0 + \boldsymbol{\Upsilon}^\intercal \boldsymbol{h}_0\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}_0^\intercal} + \frac{\partial^2 \left(\boldsymbol{\Lambda}_0^\intercal \boldsymbol{f}_0 + \boldsymbol{\Omega}_0^\intercal \boldsymbol{g}_0 + \psi + \boldsymbol{\phi}\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}_0^\intercal} \\
\boldsymbol{H}_{CN} &= \theta_N \frac{\partial^2 \left(\ell_N + \boldsymbol{\Upsilon}^\intercal \boldsymbol{h}_N\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}_N^\intercal} + \frac{\partial^2 \left(\boldsymbol{\Lambda}_N^\intercal \boldsymbol{f}_N + \boldsymbol{\Omega}_N^\intercal \boldsymbol{g}_N + \psi + \boldsymbol{\phi}\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}_N^\intercal}
\end{aligned}
\tag{6.36}
$$

As in the case of the diagonal blocks $\boldsymbol{H}_{Ak}$, the main difference between $\boldsymbol{H}_{Ck}$ for $k = 1, \ldots, N-1$ and the boundary blocks $\boldsymbol{H}_{C0}, \boldsymbol{H}_{CN}$ is the presence in the latter of the boundary conditions and Mayer target. Again, each block $\boldsymbol{H}_{Ck}$ can be written in a clearer form by substituting (6.16) into (6.36):

$$
\begin{aligned}
\boldsymbol{H}_{Ck} = &\frac{\theta_k}{2} \frac{\partial^2 \left(\ell_k + \boldsymbol{\Upsilon}^\intercal \boldsymbol{h}_k\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}^\intercal} - \frac{\partial^2 \left(\ell_k + \boldsymbol{\Upsilon}^\intercal \boldsymbol{h}_k\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}'^\intercal} \\
&+ \frac{\theta_{k-1}}{2} \frac{\partial^2 \left(\ell_{k-1} + \boldsymbol{\Upsilon}^\intercal \boldsymbol{h}_{k-1}\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}^\intercal} + \frac{\partial^2 \left(\ell_{k-1} + \boldsymbol{\Upsilon}^\intercal \boldsymbol{h}_{k-1}\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}'^\intercal} \\
&+ \frac{1}{2} \frac{\partial^2 \left(\boldsymbol{\Lambda}_k^\intercal \boldsymbol{f}_k + \boldsymbol{\Omega}_k^\intercal \boldsymbol{g}_k\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}^\intercal} - \frac{1}{\theta_k} \frac{\partial^2 \left(\boldsymbol{\Lambda}_k^\intercal \boldsymbol{f}_k + \boldsymbol{\Omega}_k^\intercal \boldsymbol{g}_k\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}'^\intercal} \\
&+ \frac{1}{2} \frac{\partial^2 \left(\boldsymbol{\Lambda}_{k-1}^\intercal \boldsymbol{f}_{k-1} + \boldsymbol{\Omega}_{k-1}^\intercal \boldsymbol{g}_{k-1}\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}^\intercal} + \frac{1}{\theta_{k-1}} \frac{\partial^2 \left(\boldsymbol{\Lambda}_{k-1}^\intercal \boldsymbol{f}_{k-1} + \boldsymbol{\Omega}_{k-1}^\intercal \boldsymbol{g}_{k-1}\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}'^\intercal}
\end{aligned}
\tag{6.37a}
$$

$$
\begin{aligned}
\boldsymbol{H}_{C0} = &\frac{\theta_0}{2} \frac{\partial^2 \left(\ell_0 + \boldsymbol{\Upsilon}^\intercal \boldsymbol{h}_0\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}^\intercal} + \frac{1}{2} \frac{\partial^2 \left(\psi + \boldsymbol{\gamma}^\intercal \boldsymbol{\phi}\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}_i^\intercal} - \frac{\partial^2 \left(\ell_0 + \boldsymbol{\Upsilon}^\intercal \boldsymbol{h}_0\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}'^\intercal} - \frac{1}{\theta_0} \frac{\partial^2 \left(\psi + \boldsymbol{\gamma}^\intercal \boldsymbol{\phi}\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}_i'^\intercal} \\
&+ \frac{1}{2} \frac{\partial^2 \left(\boldsymbol{\Lambda}_0^\intercal \boldsymbol{f}_0 + \boldsymbol{\Omega}_0^\intercal \boldsymbol{g}_0\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}^\intercal} - \frac{1}{\theta_0} \frac{\partial^2 \left(\boldsymbol{\Lambda}_k^\intercal \boldsymbol{f}_0 + \boldsymbol{\Omega}_0^\intercal \boldsymbol{g}_0\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}'^\intercal}
\end{aligned}
\tag{6.37b}
$$

$$
\begin{aligned}
\boldsymbol{H}_{CN} = &\frac{\theta_{N-1}}{2} \frac{\partial^2 \left(\ell_{N-1} + \boldsymbol{\Upsilon}^\intercal \boldsymbol{h}_{N-1}\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}^\intercal} + \frac{1}{2} \frac{\partial^2 \left(\psi + \boldsymbol{\gamma}^\intercal \boldsymbol{\phi}\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}_f^\intercal} \\
&+ \frac{\partial^2 \left(\ell_{N-1} + \boldsymbol{\Upsilon}^\intercal \boldsymbol{h}_{N-1}\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}'^\intercal} + \frac{1}{\theta_{N-1}} \frac{\partial^2 \left(\psi + \boldsymbol{\gamma}^\intercal \boldsymbol{\phi}\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}_f'^\intercal} \\
&+ \frac{1}{2} \frac{\partial^2 \left(\boldsymbol{\Lambda}_{N-1}^\intercal \boldsymbol{f}_{N-1} + \boldsymbol{\Omega}_{N-1}^\intercal \boldsymbol{g}_{N-1}\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}^\intercal} + \frac{1}{\theta_{N-1}} \frac{\partial^2 \left(\boldsymbol{\Lambda}_k^\intercal \boldsymbol{f}_{N-1} + \boldsymbol{\Omega}_{N-1}^\intercal \boldsymbol{g}_{N-1}\right)}{\partial \boldsymbol{p} \partial \boldsymbol{v}'^\intercal}
\end{aligned}
\tag{6.37c}
$$

Finally, the block $\boldsymbol{H}_D$ is simply the Hessian of the Lagrangian function (6.32) restricted to the NLP (and OCP) parameters $\boldsymbol{y}_{N+1} = \boldsymbol{p}$:

$$\boldsymbol{H}_D = \frac{\partial^2}{\partial \boldsymbol{p}^2} \left(\psi + \sum_{k=0}^{N-1} \ell_k \theta_k + \sum_{k=0}^{N-1} \left(\boldsymbol{\Lambda}_k^\intercal \boldsymbol{f}_k + \boldsymbol{\Omega}_k^\intercal \boldsymbol{g}_k \right) + \boldsymbol{\Upsilon}^\intercal \sum_{k=0}^{N-1} \left(\boldsymbol{h}_k \theta_k \right) + \boldsymbol{\gamma}^\intercal \boldsymbol{\phi} \right) \tag{6.38a}$$

6.3.3 Relationships between OCP and NLP Lagrange multipliers and necessary conditions

In optimal control problems the Lagrange multipliers related to the first-order necessary conditions (see section 2.3.1) have a relevant physical meaning. Such Lagrange multipliers indeed express the rate of change of the target with respect to the violation of the constraints to which they are associated, and in several applications it is important to know such Lagrange multipliers. However when using a direct approach to solve the OCP only the NLP multipliers (2.24) are known[7], and the OCP multipliers can only be estimated. The estimated OCP multipliers are of great importance not only for their physical meaning, but also because they can be used to check how accurately the NLP solution satisfies the optimality conditions and co-equations (see section 2.3.1). For these reasons, an OCP to NLP direct transcription method should give an estimation of the OCP Lagrange multipliers.

In this section the relationship between the OCP and NLP Lagrange multipliers is derived when using the discretisation scheme described in section 6.3.1. First, the equations that characterise the NLP necessary conditions are described, then the relationships between the OCP and NLP necessary conditions and Lagrange multipliers are derived.

As described in section 2.3.2, an NLP solution is characterised not only by the local minimum $\boldsymbol{x}^*$, but also by a set of Lagrange multipliers that satisfy the KKT conditions (2.24). Using the notation adopted in the definition of the Lagrangian function (6.32), the NLP Lagrange multipliers are $\boldsymbol{\Lambda}_k$, $\boldsymbol{\Omega}_k$ (with $k = 0, \ldots, N-1$), $\boldsymbol{\Upsilon}$ and $\boldsymbol{\Gamma}$ that are associated respectively to the constraints $\boldsymbol{f}_k$ (first order equations), $\boldsymbol{g}_k$ (path constraints), $\boldsymbol{h}$ (integral constraints) and $\boldsymbol{\phi}$ (boundary conditions). Equation (2.24) states that the first order variation of the Lagrangian function with respect to the optimisation variables is zero (at the solution). For sake of conciseness, as in section 6.2.1 a single-phase version of the original problem (6.5) is considered; the extension to the multi-phase problem is straightforward. Moreover, the constraints on the states, controls and parameters (6.5h) (6.5i) (6.5j) will be disregarded since they can be thought as included in the path constraints

[7] The reader should carefully note that the OCP Lagrange multipliers are different from the NLP ones. The former are the ones that satisfy the first order necessary conditions for the OCP (see section 2.3.1), and they have an important physical meaning. The NLP Lagrange multipliers, differently, satisfy the first order necessary conditions for the NLP problem, i.e. for a large number of algebraic equations (see section 2.3.2).

(6.5e). The first order necessary conditions of this simplified NLP problem are:

$$\mathbf{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{y}_0} = \frac{\partial \psi}{\partial \boldsymbol{y}_0} + \theta_0 \frac{\partial \ell_0}{\partial \boldsymbol{y}_0} + \boldsymbol{\Lambda}_0^{\mathsf{T}} \frac{\partial \boldsymbol{f}_0}{\partial \boldsymbol{y}0} + \boldsymbol{\Omega}_0^{\mathsf{T}} \frac{\partial \boldsymbol{g}_0}{\partial \boldsymbol{y}_0} + \theta_0 \boldsymbol{\Upsilon}^{\mathsf{T}} \frac{\partial \boldsymbol{h}_0}{\partial \boldsymbol{y}_0} + \boldsymbol{\Gamma}^{\mathsf{T}} \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{y}_0} \tag{6.39a}$$

$$\begin{aligned} \mathbf{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{y}_k} =& \theta_{k-1} \frac{\partial \ell_{k-1}}{\partial \boldsymbol{y}_k} + \theta_k \frac{\partial \ell_k}{\partial \boldsymbol{y}_k} + \boldsymbol{\Lambda}_{k-1}^{\mathsf{T}} \frac{\partial \boldsymbol{f}_{k-1}}{\partial \boldsymbol{y}_k} + \boldsymbol{\Lambda}_k^{\mathsf{T}} \frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{y}_k} \\ &+ \boldsymbol{\Omega}_{k-1}^{\mathsf{T}} \frac{\partial \boldsymbol{g}_{k-1}}{\partial \boldsymbol{y}_k} + \boldsymbol{\Omega}_k^{\mathsf{T}} \frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{y}_k}, \qquad \text{for } k = 1, \dots, N-1 \end{aligned} \tag{6.39b}$$

$$\begin{aligned} \mathbf{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{y}_N} =& \frac{\partial \psi}{\partial \boldsymbol{y}_{N-1}} + \theta_{N-1} \frac{\partial \ell_{N-1}}{\partial \boldsymbol{y}_{N-1}} + \boldsymbol{\Lambda}_{N-1}^{\mathsf{T}} \frac{\partial \boldsymbol{f}_{N-1}}{\partial \boldsymbol{y}N-1} + \boldsymbol{\Omega}_{N-1}^{\mathsf{T}} \frac{\partial \boldsymbol{g}_{N-1}}{\partial \boldsymbol{y}_{N-1}} \\ &+ \theta_{N-1} \boldsymbol{\Upsilon}^{\mathsf{T}} \frac{\partial \boldsymbol{h}_{N-1}}{\partial \boldsymbol{y}_{N-1}} + \boldsymbol{\Gamma}^{\mathsf{T}} \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{y}_{N-1}} \end{aligned} \tag{6.39c}$$

$$\begin{aligned} \mathbf{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{p}} =& \frac{\partial \psi}{\partial \boldsymbol{y}_{N+1}} + \boldsymbol{\Gamma}^{\mathsf{T}} \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{y}_{N+1}} \\ &+ \sum_{k=0}^{N-1} \left(\theta_k \frac{\partial \ell_k}{\partial \boldsymbol{y}_{N+1}} + \boldsymbol{\Lambda}_k^{\mathsf{T}} \frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{y}_{N+1}} + \boldsymbol{\Omega}_k^{\mathsf{T}} \frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{y}_{N+1}} + \theta_k \boldsymbol{\Upsilon}^{\mathsf{T}} \frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{y}_{N+1}} \right) \end{aligned} \tag{6.39d}$$

It will be now shown how equations (6.39) are closely tied to the OCP necessary conditions (6.10); this relationship will be used to express the OCP multipliers as function of the NLP ones. For this purpose it is more convenient to express equations (6.39) using derivatives calculated not with respect to the NLP variables, but with respect to the OCP variables. Thus, using the inverse of relationship (6.16), they become:

$$\mathbf{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{y}_0} = \frac{\partial \psi}{\partial \boldsymbol{v}_i} + \frac{\theta_0}{2} \frac{\partial \ell_0}{\partial \boldsymbol{v}} - \frac{\partial \ell_0}{\partial \boldsymbol{v}'} + \boldsymbol{\Lambda}_0^{\mathsf{T}} \left(\frac{1}{2} \frac{\partial \boldsymbol{f}_0}{\partial \boldsymbol{v}} - \frac{1}{\theta_0} \frac{\partial \boldsymbol{f}_0}{\partial \boldsymbol{v}'} \right) + \boldsymbol{\Omega}_0^{\mathsf{T}} \left(\frac{1}{2} \frac{\partial \boldsymbol{g}_0}{\partial \boldsymbol{v}} - \frac{1}{\theta_0} \frac{\partial \boldsymbol{g}_0}{\partial \boldsymbol{v}'} \right) \tag{6.40a}$$

$$\begin{aligned} \mathbf{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{y}_k} =& \frac{\theta_{k-1}}{2} \frac{\partial \ell_{k-1}}{\partial \boldsymbol{v}} - \frac{1}{2} \frac{\partial \ell_{k-1}}{\partial \boldsymbol{v}'} + \frac{\theta_k}{2} \frac{\partial \ell_k}{\partial \boldsymbol{v}} + \frac{1}{2} \frac{\partial \ell_k}{\partial \boldsymbol{v}'} \\ &+ \boldsymbol{\Lambda}_{k-1}^{\mathsf{T}} \left(\frac{1}{2} \frac{\partial \boldsymbol{f}_{k-1}}{\partial \boldsymbol{v}} + \frac{1}{\theta_{k-1}} \frac{\partial \boldsymbol{f}_{k-1}}{\partial \boldsymbol{v}'} \right) + \boldsymbol{\Lambda}_k^{\mathsf{T}} \left(\frac{1}{2} \frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{v}} - \frac{1}{\theta_k} \frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{v}'} \right) \\ &+ \boldsymbol{\Omega}_{k-1}^{\mathsf{T}} \left(\frac{1}{2} \frac{\partial \boldsymbol{g}_{k-1}}{\partial \boldsymbol{v}} + \frac{1}{\theta_{k-1}} \frac{\partial \boldsymbol{g}_{k-1}}{\partial \boldsymbol{v}'} \right) + \boldsymbol{\Omega}_k^{\mathsf{T}} \left(\frac{1}{2} \frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{v}} - \frac{1}{\theta_k} \frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{v}'} \right) \\ &+ \boldsymbol{\Upsilon}^{\mathsf{T}} \left(\frac{\theta_{k-1}}{2} \frac{\partial \boldsymbol{h}_{k-1}}{\partial \boldsymbol{v}} + \frac{1}{2} \frac{\partial \boldsymbol{h}_{k-1}}{\partial \boldsymbol{v}'} + \frac{\theta_k}{2} \frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{v}} - \frac{1}{2} \frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{v}'} \right), \quad \text{for } k = 1, \dots, N-1 \end{aligned} \tag{6.40b}$$

$$\begin{aligned}\mathbf{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{y}_N} =& \frac{\partial \psi}{\partial \boldsymbol{v}_f} + \frac{\theta_{N-1}}{2}\frac{\partial \ell_{N-1}}{\partial \boldsymbol{v}} + \frac{\partial \ell_{N-1}}{\partial \boldsymbol{v}'} + \boldsymbol{\Lambda}_{N-1}^{\mathsf{T}}\left(\frac{1}{2}\frac{\partial \boldsymbol{f}_{N-1}}{\partial \boldsymbol{v}} + \frac{1}{\theta_{N-1}}\frac{\partial \boldsymbol{f}_{N-1}}{\partial \boldsymbol{v}'}\right) \\ &+ \boldsymbol{\Omega}_{N-1}^{\mathsf{T}}\left(\frac{1}{2}\frac{\partial \boldsymbol{g}_{N-1}}{\partial \boldsymbol{v}} + \frac{1}{\theta_{N-1}}\frac{\partial \boldsymbol{g}_{N-1}}{\partial \boldsymbol{v}'}\right)\end{aligned} \tag{6.40c}$$

$$\mathbf{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{p}} = \frac{\partial \psi}{\partial \boldsymbol{p}} + \boldsymbol{\Gamma}^{\mathsf{T}}\frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{p}} + \sum_{k=0}^{N-1}\left(\theta_k \frac{\partial \ell_k}{\partial \boldsymbol{p}} + \boldsymbol{\Lambda}_k^{\mathsf{T}}\frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{p}} + \boldsymbol{\Omega}_k^{\mathsf{T}}\frac{\partial \boldsymbol{g}_k}{\partial \boldsymbol{p}} + \theta_k \boldsymbol{\Upsilon}^{\mathsf{T}}\frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{p}}\right) \tag{6.40d}$$

Summarising, equations (6.40) have been derived by equating to zero the first order variation of the NLP Lagrangian with respect to the NLP variables. Moreover they are satisfied at the NLP solution since they are optimality necessary conditions.

It will be now shown that under the following relationship between the NLP and OCP Lagrange multiplier, equations (6.40) are equivalent to equations (6.10) in the limit $N \to \infty$ (and they are an approximation for finite N):

$$\boldsymbol{\Lambda}_k \to \boldsymbol{\lambda}(\overline{\zeta}_k)\theta_k, \qquad \boldsymbol{\Omega}_k \to \boldsymbol{\omega}(\overline{\zeta}_k)\theta_k, \qquad \boldsymbol{\Gamma} \to \boldsymbol{\gamma}, \qquad \boldsymbol{\Upsilon} \to \boldsymbol{v} \tag{6.41}$$

It can be noted that the OCP ζ-independent multipliers ($\boldsymbol{\gamma}$, $\boldsymbol{v}$) are mapped directly to the corresponding OCP ones, while the ζ-dependent multipliers ($\boldsymbol{\lambda}$, $\boldsymbol{\omega}$) are mapped to the corresponding OCP multipliers evaluated at the middle of each mesh interval, multiplied by the mesh interval width. Similar results have been obained in [12] for a slightly different OC problem and using an equispaced mesh.

First of all, it can be noted that the NLP solution satisfies equations (6.10e) because they have been discretised and added to the NLP as constrains (by construction of the NLP). Moreover, it is easy to note that equation (6.40d) is the discretised version of equation (6.10d) accordingly to (6.41) and to the midpoint discretisation scheme. Similarly, equations (6.40a) and (6.40c) become equivalent respectively to equation (6.10a) and (6.10b) in the limit $\theta_0 \to 0$ and $\theta_{N-1} \to 0$. Indeed, if $\theta_0 \approx 0$ and $\theta_{N-1} \approx 0$, the terms proportional to the derivatives of ℓ_0, ℓ_{N-1}, $\boldsymbol{f}_0$, $\boldsymbol{f}_{N-1}$,$\boldsymbol{g}_0$, and $\boldsymbol{g}_{N-1}$ with respect to $\boldsymbol{v}$ tend to zero (if the value of the derivatives are bounded). Finally, it only remains to demonstrate that equations (6.40b) are the discretised version of (6.10c). This is slightly more cumbersome to show than the other equations.

Suppose to discretise equation (6.10c) not at the middle of each mesh interval, but at the dual point $\zeta_k^* = \overline{\zeta}_k - (\zeta_k - \overline{\zeta}_{k-1})$ which is located between the points $\overline{\zeta}_{k-1}$ and $\overline{\zeta}_k$ as shown in figure 6.4. Moreover, since the NLP multipliers, target and constraints are discretised at the middle of each mesh interval (accordingly to (6.41)), the equation discretised at ζ_k^* should be expressed using these NLP multipliers, target and constraints (i.e. their values

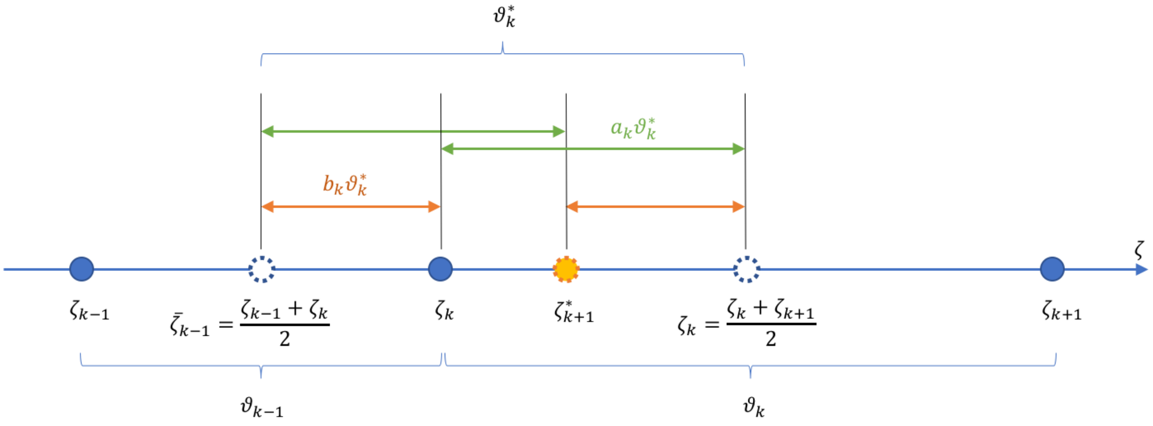


Figure 6.4: Representation of the dual point ζ_k^* (highlighted in orange) position in the mesh. ζ_k^* is located between ζ_{k-1} and ζ_k. The weighting coefficients a_k, b_k that appear in equation (6.42) are shown multiplied by the dual interval width θ_k^*.

at the middle of each mesh intervals). Following the midpoint discretisation scheme, the quantities $\frac{\partial \ell}{\partial \boldsymbol{v}}$, $\boldsymbol{\lambda}^\mathsf{T}\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{v}}$, $\boldsymbol{\omega}^\mathsf{T}\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{v}}$ and $\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{v}}$ that appear in equation (6.10c) can be estimated at the dual point ζ_k^* as follows:

$$\begin{aligned}
\frac{\partial \ell}{\partial \boldsymbol{v}}\bigg|_{\zeta_k^*} &\approx b_k \frac{\partial \ell}{\partial \boldsymbol{v}}\bigg|_{\bar{\zeta}_{k-1}} + a_k \frac{\partial \ell}{\partial \boldsymbol{v}}\bigg|_{\bar{\zeta}_k} \\
\left(\boldsymbol{\lambda}^\mathsf{T}\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{v}}\right)\bigg|_{\zeta_k^*} &\approx b_k \left(\boldsymbol{\lambda}^\mathsf{T}\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{v}}\right)\bigg|_{\bar{\zeta}_{k-1}} + a_k \left(\boldsymbol{\lambda}^\mathsf{T}\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{v}}\right)\bigg|_{\bar{\zeta}_k} \\
\left(\boldsymbol{\omega}^\mathsf{T}\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{v}}\right)\bigg|_{\zeta_k^*} &\approx b_k \left(\boldsymbol{\omega}^\mathsf{T}\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{v}}\right)\bigg|_{\bar{\zeta}_{k-1}} + a_k \left(\boldsymbol{\omega}^\mathsf{T}\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{v}}\right)\bigg|_{\bar{\zeta}_k} \\
\boldsymbol{\upsilon}^\mathsf{T}\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{v}}\bigg|_{\zeta_k^*} &\approx \boldsymbol{\upsilon}^\mathsf{T}\left(b_k \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{v}}\bigg|_{\bar{\zeta}_{k-1}} + a_k \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{v}}\bigg|_{\bar{\zeta}_k}\right)
\end{aligned} \tag{6.42}$$

where a_k and b_k are two weights that depend on the distance between ζ_k^*, $\bar{\zeta}_{k-1}$ and $\bar{\zeta}_k$ (see figure 6.4):

$$\theta_k^* = \frac{\theta_{k-1}+\theta_k}{2} \qquad a_k \;= \frac{1}{2}\frac{\theta_k}{\theta_k^*} \qquad b_k = \frac{1}{2}\frac{\theta_{k-1}}{\theta_k^*} \tag{6.43}$$

The above relationship implies that $a_k + b_k = 1$. With a_k and b_k defined as above, the expression in (6.42) are estimated at the dual point assuming they vary linearly from ζ_{k-1} to ζ_k (accordingly to the midpoint integration scheme).

Moreover, an estimate at the dual point ζ_k^* for the ζ-derivatives that appear in (6.10c) is required:

$$\begin{aligned}
\frac{d}{d\zeta}\frac{\partial \ell}{\partial \boldsymbol{v}'}\Big|_{\zeta_k^*} &\approx \frac{1}{\theta_k^*}\left(\frac{\partial \ell}{\partial \boldsymbol{v}'}\Big|_{\bar{\zeta}_k} - \frac{\partial \ell}{\partial \boldsymbol{v}'}\Big|_{\bar{\zeta}_{k-1}}\right)\\
\frac{d}{d\zeta}\left(\boldsymbol{\lambda}^{\mathsf{T}}\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{v}'}\right)\Big|_{\zeta_k^*} &\approx \frac{1}{\theta_k^*}\left[\left(\boldsymbol{\lambda}^{\mathsf{T}}\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{v}'}\right)\Big|_{\bar{\zeta}_k} - \left(\boldsymbol{\lambda}^{\mathsf{T}}\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{v}'}\right)\Big|_{\bar{\zeta}_{k-1}}\right]\\
\frac{d}{d\zeta}\left(\boldsymbol{\omega}^{\mathsf{T}}\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{v}'}\right)\Big|_{\zeta_k^*} &\approx \frac{1}{\theta_k^*}\left[\left(\boldsymbol{\omega}^{\mathsf{T}}\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{v}'}\right)\Big|_{\bar{\zeta}_k} - \left(\boldsymbol{\omega}^{\mathsf{T}}\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{v}'}\right)\Big|_{\bar{\zeta}_{k-1}}\right]\\
\boldsymbol{\upsilon}^{\mathsf{T}}\frac{d}{d\zeta}\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{v}'}\Big|_{\zeta_k^*} &\approx \frac{\boldsymbol{\upsilon}^{\mathsf{T}}}{\theta_k^*}\left(\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{v}'}\Big|_{\bar{\zeta}_k} - \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{v}'}\Big|_{\bar{\zeta}_{k-1}}\right)
\end{aligned}\tag{6.44}$$

Again, these ζ-derivatives are estimated assuming each function varies linearly between two known points (accordingly to the midpoint integration scheme).

Substituting equations (6.42) and (6.44) into (6.10c), the following expression holds:

$$\begin{aligned}
\mathbf{0} = {}& \frac{\theta_{k-1}}{2\theta_k^*}\frac{\partial \ell}{\partial \boldsymbol{v}}\Big|_{\bar{\zeta}_{k-1}} + \frac{\theta_k}{2\theta_k^*}\frac{\partial \ell}{\partial \boldsymbol{v}}\Big|_{\bar{\zeta}_k} + \frac{1}{\theta_k^*}\left(\frac{\partial \ell}{\partial \boldsymbol{v}'}\Big|_{\bar{\zeta}_k} - \frac{\partial \ell}{\partial \boldsymbol{v}'}\Big|_{\bar{\zeta}_{k-1}}\right)\\
&+ \frac{\theta_{k-1}}{2\theta_k^*}\left(\boldsymbol{\lambda}^{\mathsf{T}}\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{v}}\right)\Big|_{\bar{\zeta}_{k-1}} + \frac{\theta_k}{2\theta_k^*}\left(\boldsymbol{\lambda}^{\mathsf{T}}\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{v}}\right)\Big|_{\bar{\zeta}_k} + \frac{1}{\theta_k^*}\left(\left(\boldsymbol{\lambda}^{\mathsf{T}}\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{v}'}\right)\Big|_{\bar{\zeta}_k} - \left(\boldsymbol{\lambda}^{\mathsf{T}}\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{v}'}\right)\Big|_{\bar{\zeta}_{k-1}}\right)\\
&+ \frac{\theta_{k-1}}{2\theta_k^*}\left(\boldsymbol{\omega}^{\mathsf{T}}\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{v}}\right)\Big|_{\bar{\zeta}_{k-1}} + \frac{\theta_k}{2\theta_k^*}\left(\boldsymbol{\omega}^{\mathsf{T}}\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{v}}\right)\Big|_{\bar{\zeta}_k} + \frac{1}{\theta_k^*}\left(\left(\boldsymbol{\omega}^{\mathsf{T}}\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{v}'}\right)\Big|_{\bar{\zeta}_k} - \left(\boldsymbol{\omega}^{\mathsf{T}}\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{v}'}\right)\Big|_{\bar{\zeta}_{k-1}}\right)\\
&+ \frac{\theta_{k-1}}{2\theta_k^*}\boldsymbol{\upsilon}^{\mathsf{T}}\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{v}}\Big|_{\bar{\zeta}_{k-1}} + \frac{\theta_k}{2\theta_k^*}\boldsymbol{\upsilon}^{\mathsf{T}}\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{v}}\Big|_{\bar{\zeta}_k} + \frac{\boldsymbol{\upsilon}^{\mathsf{T}}}{\theta_k^*}\left(\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{v}'}\Big|_{\bar{\zeta}_k} - \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{v}'}\Big|_{\bar{\zeta}_{k-1}}\right)
\end{aligned}\tag{6.45}$$

Equation (6.45), multiplied by θ_k^* and using the inverse mapping of (6.41), becomes equation (6.40b).

Summarising, it has been shown that the mapping (6.41) allows to relate the OCP Lagrange multipliers and necessary conditions to the NLP ones. In particular the ζ-dependent OCP multipliers $\boldsymbol{\lambda}(\bar{\zeta}_k)$, $\boldsymbol{\omega}(\bar{\zeta}_k)$ correspond to the NLP multipliers $\boldsymbol{\Lambda}_k$, $\boldsymbol{\Omega}_k$ divided by the mesh interval width θ_k. For the ζ-independent Lagrange multipliers $\boldsymbol{\gamma}$, $\boldsymbol{\upsilon}$ there is instead a direct mapping to the NLP multipliers $\boldsymbol{\Gamma}$, $\boldsymbol{\Upsilon}$. However, it has been shown that the latter are equal to the former only in the limit $\theta_0 \to 0$, $\theta_{N-1} \to 0$, i.e. when the mesh is made enough fine. In real applications, $\boldsymbol{\Gamma}$, $\boldsymbol{\Upsilon}$ are an approximation of $\boldsymbol{\gamma}$, $\boldsymbol{\upsilon}$ that can be improved by lowering the mesh intervals width.

It can be noted that a more direct mapping between $\boldsymbol{\lambda}(\bar{\zeta}_k)$, $\boldsymbol{\omega}(\bar{\zeta}_k)$ and $\boldsymbol{\Lambda}_k$, $\boldsymbol{\Omega}_k$ could be obtained if the NLP constraints $\boldsymbol{f}_k$, $\boldsymbol{g}_k$ would be replaced respectively by $\boldsymbol{f}_k\theta_k$ and $\boldsymbol{g}_k\theta_k$. In

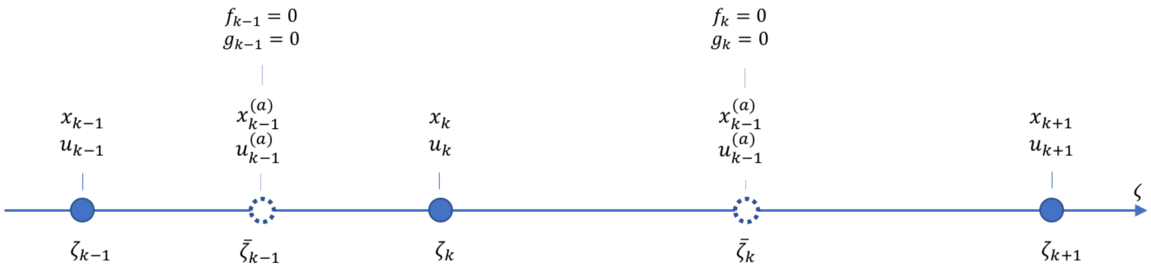


Figure 6.5: Modification of the midpoint integration scheme used for the transcription of the OCP into an NLP. The algebraic states and controls, denoted by a superscript $^{(a)}$ are discretised at the middle of each mesh interval $\overline{\zeta}_k$. The discretisation of the first order equations, constraints, and other (differential) states and controls is not affected.

such case, the mapping (6.41) would become:

$$\boldsymbol{\Lambda}_k \to \boldsymbol{\lambda}(\overline{\zeta}_k), \qquad \boldsymbol{\Omega}_k \to \boldsymbol{\omega}(\overline{\zeta}_k), \qquad \boldsymbol{\Gamma} \to \boldsymbol{\gamma}, \qquad \boldsymbol{\Upsilon} \to \boldsymbol{\upsilon} \tag{6.46}$$

Finally, it should be considered that the close relationship between the OCP necessary conditions and the NLP ones does not mean that the two approaches are exactly equivalent. Indeed, if equations (6.10) are discretised and solved, the resulting finite dimensional algebraic equation system would be slightly different than (6.40) (plus the constraint equations). The two approaches become closer and closer in the limit $N \to \infty$ and $\theta_k \to 0$.

6.3.4 Notes on algebraic variables

The discretisation scheme proposed and used in the previous sections, where both states and controls are discretised at the boundaries and the equations at the middle of each mesh interval, has been motivated by the need for having available control derivatives in the optimal problem formulation. There are optimal control problems where the controls, and some states, are algebraic variables and their derivatives do not appear in the first order equations. For this class of problems the discretisation scheme requires a slight modification in order to avoid numerical issues. If the same discretisation method proposed in section 6.3.1 is used for the algebraic variables, then on a mesh with $N + 1$ discretisation points, every algebraic state or control is discretised into $N+1$ discrete variables, one on each mesh point. Moreover there are only in N equations ($\boldsymbol{f}_0,\ldots,\boldsymbol{f}_{N-1}$), one per each mesh interval, where they take part, thus the problem is under-determined. This issue can be avoided by discretising the algebraic states and controls not on each mesh point ζ_k, but on each mesh interval middle point $\overline{\zeta}_k$, as shown in figure 6.5. The discretisation of the first order equations, constraints, and other (differential) states and controls is not modified.

The change in the resulting NLP structure discussed in section 6.3.2 is only marginal, and is not described here.

Boundary conditions that depend on algebraic variables need particular attention, since they not only need to satisfy the algebraic constraints, but also other hidden constraints [38, 39]. With the proposed discretisation scheme for the algebraic variables and constraints, these issues are outflanked since boundary conditions cannot depend on such variable, which are not discretised at the exact boundary of the domain, i.e. ζ_i, ζ_f.

Finally, it is easy to verify that the relationship between the NLP and the OCP Lagrange multipliers for the algebraic variables is the same as in (6.41) — or (6.46) if the NLP equations are multiplied by θ_k.

6.3.5 Software implementation

The full collocation transcription method for implicit OCPs described in the previous pages has been implemented in a software called *Maverick* that is available on *GitHub* at `http://github.com/stavoltafunzia/Maverick`. *Maverick* is composed of a *Maple*©[8] library that allows to easily declare the optimal control problem (variables, equations, constraints, boundary conditions and target) in a symbolic algebra environment, and of a core written in *C++* for the translation of the OCP to an NLP problem. All data necessary to solve the OCP, like the numeric dataset, the mesh to use, the problem scaling and the solver settings, can be declared from a user friendly *Python* and/or *Lua* interface.

The main features of the *Maverick* software are the following:

- it allows to easily declare the OCP from within the *Maple*© symbolic algebra software;
- it automatically generate the first and second order derivatives of the OCP functions;
- it automatically generate the *C++* code for the OCP functions;
- user-friendly interface to control the solver workflow and for the declaration of the OCP dataset, mesh, scaling, and solver settings;
- it automatically convert the OCP to NLP problem (including Jacobian and Hessian matrices);
- it automatically refine the mesh to satisfy the requested equation integration accuracy.

The workflow of the *Maverick* software is shown in figure 6.6. The core of the software is entirely written in *C++* so as to provide very fast function evaluations and to reduce

[8] Maple is a symbolic algebra software

computing times of the solution. The core of the software acts as a junction between the OCP and the NLP problem; the OCP functions, that are coded in *C++*, are evaluated on the desired mesh and the corresponding NLP problem is built, including target, constraints and Jacobian and Hessian matrices. The evaluation is performed on multiple threads to exploit multi-core architecture. The NLP problem is then solved by third party NLP solvers, like *IPOPT* [95]. The OCP dataset, together with the desired mesh, OCP scaling, and solver settings, is declared in *Python* and/or *Lua* languages and passed to the *C++* core through dedicated interfaces. The OCP *C++* code, even if it can be manually written, is automatically generated from within the *Maple©* software thanks to the *Maverick-Maple©* library. This allows the user to easily declare the OCP variables, equations, constraints, boundary conditions and target in the symbolic algebra environment without caring of calculating the functions derivatives nor writing a single line of *C++* code. The calculus of OCP functions derivatives and the transcription into *C++* code is automatically done by the *Maverick-Maple©* library. The *Maple©* environment becomes even more appealing and easy to use for the OCP declaration when the *MBSymba* library is used. *MBSymba* library [60] is a tool that automatically generates the equations of motion for a dynamical system after the uses specifies the relevant reference systems, bodies and forces in a user-friendly way. Finally *Maverick* can automatically refine the mesh if the equation integration accuracy is not satisfied on the current mesh. This is achieved by estimating the integration accuracy on each mesh interval, and splitting the interval into sub-intervals [9] if the estimated accuracy is below the user-selected threshold.

Maverick software has been used to solve the optimal control problem described in the next section; such problem consists in finding the minimum manoeuvre time of a motorbike performing a U-turn on a road characterised by relevant elevation variations. The software is currently in used in the research group at the University of Padova in which the author worked during his PhD.

6.4 Application to 3D road-vehicle models

Among the applications where the implicit optimal control formulation is advantageous over the classic explicit formulation, there is the problem arising from a vehicle with 6 degrees of freedom (for the chassis motion) moving on a three dimensional road. In this section such OCP is first presented, then it will be solved for a motorbike executing a 180-degrees U-turn on a three-dimensional road [10] . A thorough comparison of *Maverick* and of the implicit

[9] The number of intervals in which a single mesh interval is split is calculated with an heuristic algorithm.

[10] The term “three-dimensional” road is here used improperly since the road actually is a two-dimensional manifold. Three-dimensional means that the road is not flat and it extends on a three-dimensional space,

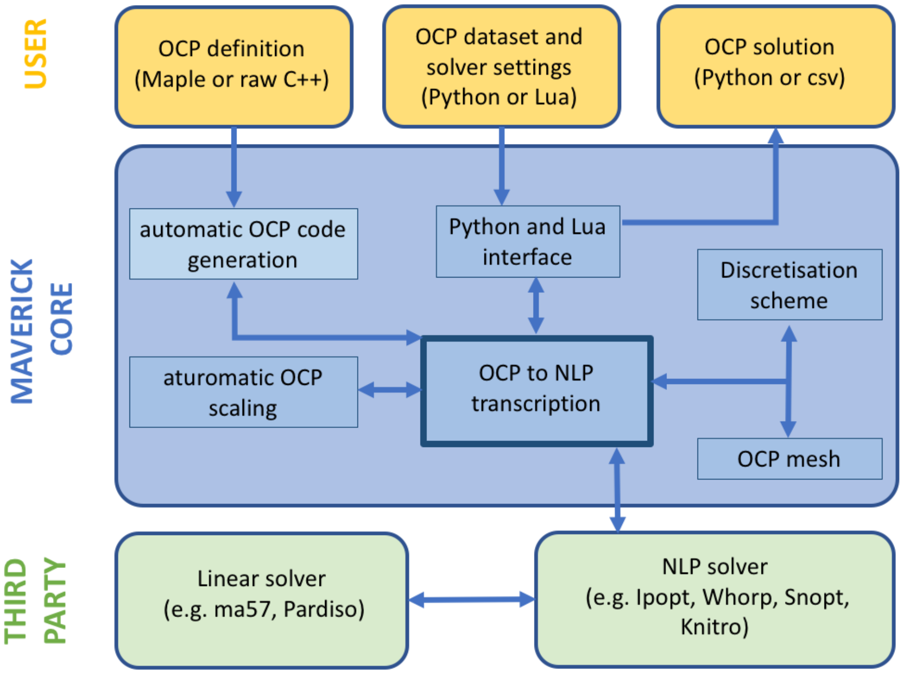

USER
OCP definition (Maple or raw C++)
OCP dataset and solver settings (Python or Lua)
OCP solution (Python or csv)
MAVERICK CORE
automatic OCP code generation
Python and Lua interface
Discretisation scheme
aturomatic OCP scaling
OCP to NLP transcription
OCP mesh
THIRD PARTY
Linear solver (e.g. ma57, Pardiso)
NLP solver (e.g. Ipopt, Whorp, Snopt, Knitro)

Figure 6.6: Overview of the *Maverick* software structure. The user can declare the OCP in the symbolic algebra environment *Maple*©, then the OCP dataset and all solver settings can be specified through *Python* and/or *Lua* interfaces. *Maverick* software generates the *C++* code for the OCP and converts it to an NLP problem that is then solved by a (free) third party library.

formulation with respect to other existing OCP software and the explicit formulation is out of the scope of this chapter, which aims only to give some kind of "proof of work" of the developed software.

A three-dimensional road for optimal control problems of vehicles can be modelled as in appendix A. Following the road model there described, the vehicle frame $\boldsymbol{W}_v$ is related to the road one $\boldsymbol{W}_r$ through the following transformation:

$$\boldsymbol{W}_v(t) = \boldsymbol{W}_r(s(t))\boldsymbol{W}_{rel}(\boldsymbol{x}_{rel}(t)) \tag{6.47}$$

where $\boldsymbol{W}_{rel}$ is the relative transformation frame that depends on the relative degrees of freedom $\boldsymbol{x}_{rel}$. In the general case, a vehicle has six dof (related to the chassis) thus $\boldsymbol{x}_{rel}$ includes five variables, three of which are related to the rotations about the three axis and two of which are related to translations (one perpendicular to the road plane and one perpendicular to the road middle lane). The degree of freedom associated with the remaining translation along the road middle lane, $s(t)$, is already included in the $\boldsymbol{W}_r$ frame. The rotations and translations can be performed in different orders, leading to slightly different meaning of the five dof $\boldsymbol{x}_{rel}$. Here the $\boldsymbol{W}_{rel}$ matrix is given by the following transformation sequence: a displacement of n along the y-axis, followed by a rotation of α about the z-axis, a rotation of ϕ about the x-axis, a translation of $-h+z$ along the z-axis (h is the CoM nominal height from ground) and finally a rotation of μ about the y-axis. Thus the $\boldsymbol{W}_{rel}$ matrix is:

$$\boldsymbol{W}_r(s) = \begin{bmatrix} \cos\mu\cos\alpha - \sin\phi\sin\alpha\sin\mu & -\cos\phi\sin\alpha & \sin\mu\cos\alpha + \sin\phi\sin\alpha\cos\mu & -(h-z)\sin\psi\sin\alpha \\ \cos\mu\sin\alpha + \sin\phi\cos\alpha\sin\mu & \cos\phi\cos\alpha & \sin\mu\sin\alpha - \sin\phi\cos\alpha\cos\mu & n + (h-z)\sin\phi\cos\alpha \\ -\sin\mu\cos\phi & \sin\phi & \cos\mu\cos\phi & -(h-z)\cos\phi \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.48}$$

Thus n is the lateral displacement from the road middle lane, α is the heading angle w.r.t the road middle lane, ϕ is the roll angle, z is the vertical displacement from the nominal CoM height and μ is the pitch angle. These five variables, together with the position of the vehicle along the road middle lane s, completely determine the position and orientation of the vehicle.

The equations of motion for the vehicle gross motion requires 12 variables in order to be described by a set of first order equations. Therefore, in addition to the six position position variables s, n, h, ϕ, μ and α, six more (velocity) variables are needed. The additional six velocity variables can be arbitrarily chosen as long as they are linearly independent; for instance they may be the time derivative of the position variables. However Newton and Euler equations can be written in a very compact form if the velocities of the vehicle frame

i.e. the road presents elevation variations.

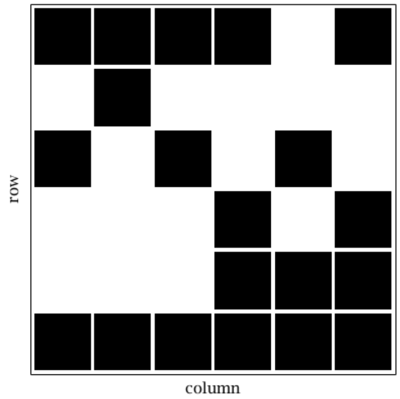


Figure 6.7: Sparse structure of the matrix $\boldsymbol{A}$ in equation (6.50).

$\boldsymbol{v}_v$ and $\boldsymbol{\omega}_v$ are used as the additional six state variables. Indeed, with this choice the Newton and Euler equations for the whole vehicle become:

$$\begin{aligned} \boldsymbol{F} &= m_v(\dot{\boldsymbol{v}}_v + \boldsymbol{\omega}_v \times \boldsymbol{v}_v) \\ \boldsymbol{\tau} &= \boldsymbol{I}_v\dot{\boldsymbol{\omega}}_v + \boldsymbol{\omega}_v \times \boldsymbol{I}_v\boldsymbol{\omega}_v \end{aligned} \tag{6.49}$$

where $\boldsymbol{F}$ and $\boldsymbol{\tau}$ are respectively the sum of the net external forces and torques acting on the vehicle, m_v and $\boldsymbol{I}_v$ are the vehicle mass and inertia tensor. Summarising, with this choice the state space model includes the following variables: s, n, h, ϕ, μ, α, v_{vx}, v_{vy}, v_{vz}, ω_{vx}, ω_{vy}, ω_{vz}.

While the explicit Newton and Euler equations of motion (6.49) depend on the vehicle model used (the external forces and torques depend on the vehicle model), the other six first order equations are independent of the vehicle model: these equations relate the rate of the position variables to the vehicle frame velocities $\boldsymbol{v}_v$, $\boldsymbol{\omega}_v$. Such equations, that can be derived through kinematics considerations, include only the state space variables, the time derivatives of the six position variables, and the road curvatures:

$$\begin{bmatrix} v_{vx} \\ v_{vy} \\ v_{vz} \\ \omega_{vx} \\ \omega_{vy} \\ \omega_{vz} \end{bmatrix} = \boldsymbol{A} \begin{bmatrix} \dot{s} \\ \dot{n} \\ \dot{z} \\ \dot{\phi} \\ \dot{\mu} \\ \dot{\alpha} \end{bmatrix} \tag{6.50}$$

where v_{vx}, v_{vy}, v_{vz}, ω_{vx}, ω_{vy}, ω_{vz} are the components of the vectors $\boldsymbol{v}_v$, $\boldsymbol{\omega}_v$, and the matrix

$\boldsymbol{A}$ has the following non-zero entries:

$$
\begin{aligned}
\boldsymbol{A}_1^1 =&(h-z)\cos(\phi)\sin(\alpha)\cos(\mu)\tau - (h-z)\cos(\phi)\cos(\mu)\cos(\alpha)\nu \\
&+ \sin(\alpha)\sin(\phi)n\sin(\mu)\kappa - (h-z)\cos(\mu)\sin(\phi)\kappa \\
&- \cos(\phi)\tau n\sin(\mu) - \cos(\mu)\cos(\alpha)n\kappa - \sin(\alpha)\sin(\phi)\sin(\mu) + \cos(\mu)\cos(\alpha) \\
\boldsymbol{A}_1^2 =& \cos(\alpha)\sin(\phi)\sin(\mu) + \sin(\alpha)\cos(\mu) \\
\boldsymbol{A}_1^3 =& \sin(\mu), \quad \boldsymbol{A}_1^5 = -(h-z)\cos(\mu)\sin(\phi) \\
\boldsymbol{A}_2^1 =& \cos(\phi)\sin(\alpha)n\kappa + (h-z)\sin(\alpha)\nu + (h-z)\tau\cos(\alpha) \\
&+ \tau\sin(\phi)n - \cos(\phi)\sin(\alpha) \\
\boldsymbol{A}_2^2 =& \cos(\phi)\cos(\alpha), \quad \boldsymbol{A}_2^4 = h \\
\boldsymbol{A}_3^1 =&(h-z)\cos(\phi)\sin(\alpha)\tau\sin(\mu) - (h-z)\cos(\phi)\cos(\alpha)\nu\sin(\mu) \\
&- \sin(\alpha)\cos(\mu)\sin(\phi)n\kappa - (h-z)\sin(\phi)\sin(\mu)\kappa \\
&+ \cos(\phi)\cos(\mu)\tau n - \cos(\alpha)n\sin(\mu)\kappa + \sin(\alpha)\cos(\mu)\sin(\phi) + \cos(\alpha)\sin(\mu) \\
\boldsymbol{A}_3^2 =& -\cos(\mu)\cos(\alpha)\sin(\phi) + \sin(\alpha)\sin(\mu) \\
\boldsymbol{A}_3^3 =& -\cos(\mu), \quad \boldsymbol{A}_3^6 = -(h-z)\sin(\phi)\sin(\mu) \\
\boldsymbol{A}_4^1 =& -\sin(\alpha)\tau\sin(\phi)\sin(\mu) + \cos(\alpha)\nu\sin(\phi)\sin(\mu) - \cos(\phi)\sin(\mu)\kappa \\
&+ \sin(\alpha)\cos(\mu)\nu + \cos(\mu)\tau\cos(\alpha) \\
\boldsymbol{A}_4^4 =& \cos(\mu), \quad \boldsymbol{A}_4^6 = -\cos(\phi)\sin(\mu) \\
\boldsymbol{A}_5^1 =& -\cos(\phi)\sin(\alpha)\tau + \cos(\phi)\cos(\alpha)\nu + \sin(\phi)\kappa, \quad \boldsymbol{A}_5^5 = 1, \quad \boldsymbol{A}_5^6 = \sin(\phi) \\
\boldsymbol{A}_6^1 =& \sin(\alpha)\cos(\mu)\tau\sin(\phi) - \cos(\mu)\cos(\alpha)\nu\sin(\phi) + \cos(\phi)\cos(\mu)\kappa \\
&+ \sin(\alpha)\nu\sin(\mu) + \tau\cos(\alpha)\sin(\mu) \\
\boldsymbol{A}_6^4 =& \sin(\mu), \quad \boldsymbol{A}_6^6 = \cos(\phi)\cos(\mu)
\end{aligned}
\tag{6.51}
$$

The resulting sparse structure of the matrix $\boldsymbol{A}$ is shown in figure 6.7. Since the matrix $\boldsymbol{A}$ is not diagonal, these equations are not explicit. It comes thus natural to use an implicit OCP formulation. If an explicit OCP formulation is required, the equations must be written in explicit form. This could be done by inverting the matrix $\boldsymbol{A}$, but it would make the equations more complex. Moreover, if the vehicle Newton and Euler equations include the derivative of the position variables (as it is in the motorbike model that is used here), then the six equations in (6.50) cannot be made explicit separately from the equations of motion (6.49). Thus, it is the mass matrix of the whole 12 equations that has to be inverted, and this may not be doable.

Table 6.1: Motorcycle numeric dataset used for the U-turn manoeuvre simulation.

variable	value	units	description
m	300	kg	total mass
w	1.4	m	wheelbase
b	0.6	b	rear wheel to CoM longitudinal distance
I_{xx}	40	kgm^2	total inertia moment about the x-axis
I_{zz}	60	kgm^2	total inertia moment about the y-axis
I_{xx}	40	kgm^2	total inertia moment about the z-axis
$I_{wf} = I_{wr}$	1	kgm^2	rear and front wheel spin inertia
I_e	1	kgm^2	engine spin inertia reduced at wheel
$r_t = r_{tr}$	0.3	m	tyre nominal and rolling radius
$\mu_x^{(r)} = \mu_x^{(f)}$	1	$-$	tyre maximum longitudinal adherence
$\mu_y^{(r)} = \mu_y^{(f)}$	1	$-$	tyre maximum lateral adherence
$K_{t\lambda}^{(r)}$	10	$1/rad$	rear tyre sideslip stiffness
$K_{t\phi}^{(r)}$	10	$1/rad$	rear tyre roll stiffness
$K_{er}^{(r)} = K_{er}^{(f)}$	20×10^3	N/m	equivalent radial stiffness
$K_{ed}^{(r)} = K_{ed}^{(f)}$	1×10^3	Ns/m	equivalent damping
c_{dA}	0.5	m^2	drag coefficient
c_{lA}	0	m^2	lift coefficient

6.4.1 3D U-turn motorcycle example

In this section the minimum time problem of a motorbike that performs a 180-degrees U-turn on a $3D$ road is presented.

The motorbike model is composed by the main chassis plus the two wheels. The detailed description of the motorbike model is not of primary importance here and can be found in appendix C. What is here relevant is that the Newton and Euler equations, as described in appendix C, are neither explicit nor linear in the derivatives of the state space variables. Indeed, the tyre lateral force expression includes some terms that are quadratic in the derivatives of the position variables. Thus, the equations cannot be straightforwardly written neither in the classic explicit form (6.1) nor using the mass matrix (6.2); thus the implicit OCP formulation is of great advantage here. The minimum manoeuvre time problem is here solved with the implicit OCP to NLP transcription procedure described in this chapter and implemented in the *Maverick* software; *IPOPT* has been used as NLP solver. The dataset used to feed the model is reported in table 6.1.

The $3D$ road is composed of a flat straight line $100m$ long, followed by a 180-degrees U-turn approximately $200m$ long (curvature radius of $\approx 63m$) and a final flat straight $50m$ long. The overall elevation variation is of approximately 20 meters and it is entirely comprised in the U-turn section, with an average slope along the road middle lane of $12°$

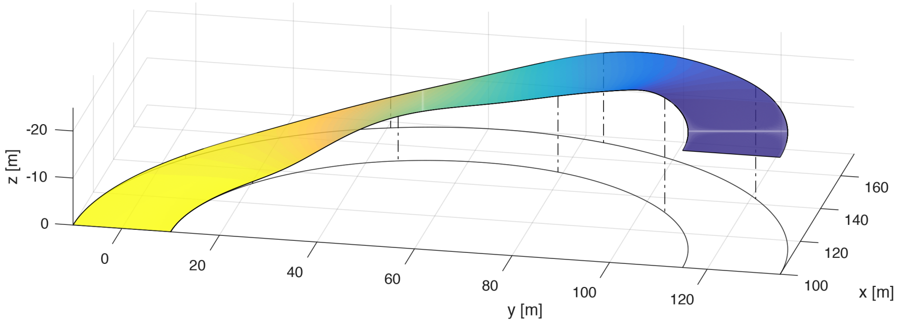


Figure 6.8: Three-dimensional overview of the 180° U-turn (“up hill”).

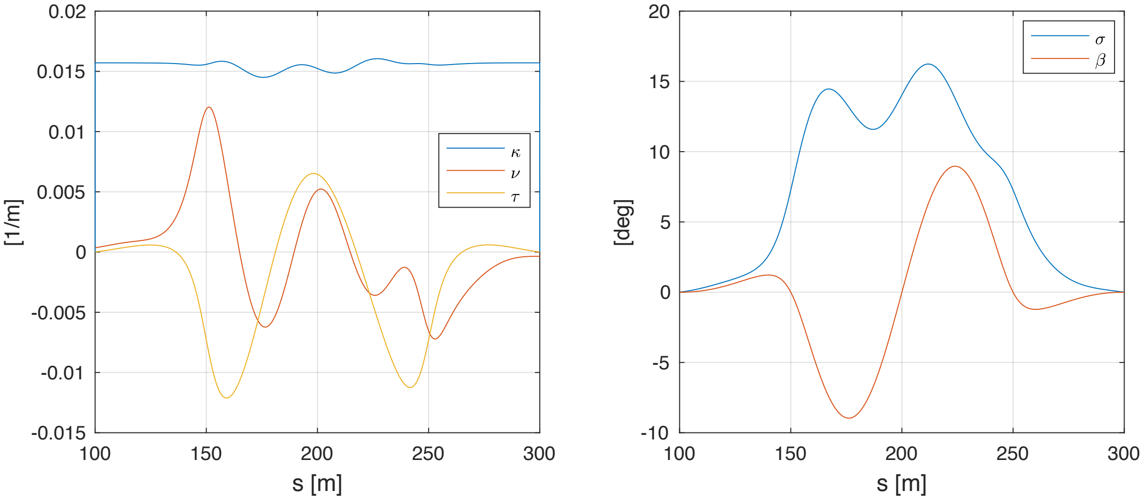


Figure 6.9: Road curvatures (left) and tilting angles (road) along the 180° U-turn (“up hill”).

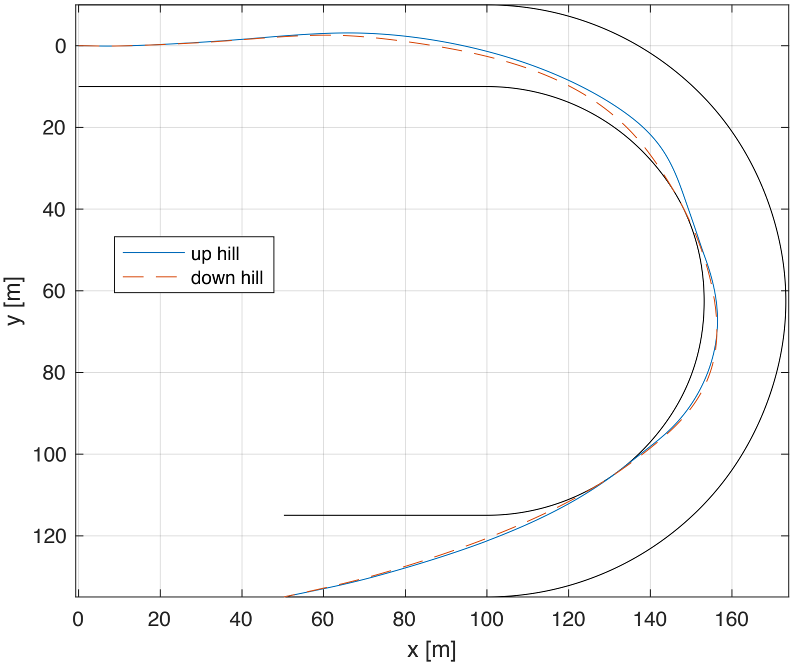


Figure 6.10: Trajectory of the motorcycle. The difference between the “up hill” and “down hill” roads is noticeable. The maximum distance between the two trajectories is of $\approx 3m$.

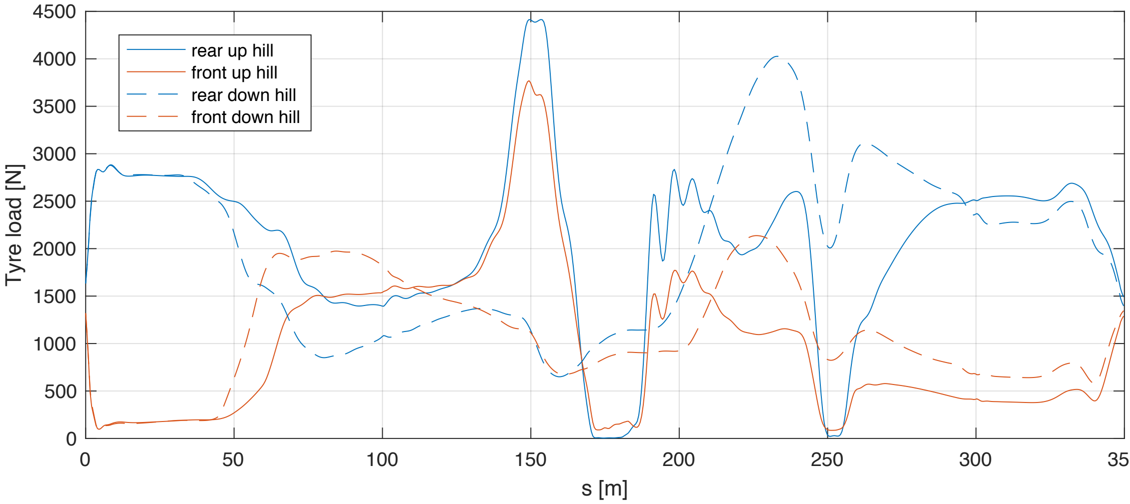


Figure 6.11: Tyre loads along the 180° U-turn . Bump effects are very clear for the “up hill” motorcycle at $s \approx 150m$ and $s \approx 240m$: tyre loads first increase (at $s \approx 150$ both tyre loads are greater than the motorcycle weight) and then diminish to almost zero. In the “down hill” road a small bump can be observed at $s \approx 240m$.

and a maximum one of the 29°. A $3D$ overview of the road is shown in figure 6.8. The road curvatures, together with the slope and banking angles, are shown in figure 6.9. The κ curvature is almost constant along the turn, with a value of $\approx 0.015m^{-1}$, while the other curvatures ν and κ have an oscillating trend. The road slope σ increases from zero up to $\approx 20°$ and then decreases back to zero, moreover the banking angle oscillates between positive and negative values, being always less than 10° in magnitude.

In order to highlight the influence of the road geometry on the U-turn manoeuvre, the manoeuvre has been simulated both on the road just described, and on the same road with the opposite elevation, i.e. with opposite slope and banking angles. The original road will be referred to as "up hill", the latter as "down hill". The simulated manoeuvre time is of $11.727s$ for the "up hill" road, and $11.642s$ for the "down hill" one. Significant differences in the motorbike trajectory can be observed between the two roads, as shown in figure 6.10. It can be noted that in the first part of the turn the "up hill" trajectory is farther from the interior of the turn, and the maximum distance between the two trajectories is slightly greater than $3m$. The different trajectory choice are a consequence of the different road slope variations, that leads to different loads on the tyres. Tyre loads are shown in figure 6.11: it is evident that at $s \approx 150m$ the motorcycle in the "up hill" road experiences a significant bump that makes the tyre loads to significantly increase first (both front and rear tyre loads are higher than the motorcycle weight), and then to become almost zero. The same behaviour can be noticed also at $s \approx 240m$. The motorcycle in the "down hill" road instead presents smoother tyre loads, yet a small bump can be observed at $s \approx 240m$.

The example here described has shown that the implicit OCP formulation can be advantageous in some applications, including OC simulations of vehicles running on $3D$ roads. Moreover, the influence of road slope variations on the motorcycle manoeuvre has been shown to affect both the motorcycle trajectory and tyre loads.

6.5 Summary

In this chapter optimal control problems with implicit first order dynamics equations have been studied. It has been shown that implicit dynamics equations are of great advantage from a modelling point of view since they allow to more easily develop the mathematical model of dynamical systems. Moreover, in some scenarios, writing explicit first order equations may be really cumbersome. It is not so unusual to find dynamical systems where the implicit formulation is advantageous; indeed the equations of motion of mechanical systems are intrinsically implicit[11], and not always they can be straightforwardly written in explicit

[11] The equations of motion of mechanical systems are generally linear in the state derivatives, thus they can be written as in equation (6.2). However, sometimes it may be not straightforwardly written in such

form. Moreover some dynamical systems may be intrinsically described by functions that are implicit in the state derivatives, as in the muscular activation dynamics problem treated in [31].

In addition to the implicit first order equations, the OCP formulation here presented allows for the control derivatives to appear in the expressions of the OCP target, constraints and equations. This gives even more freedom from a modelling point of view, since it allows to add constraints or regularisation terms related to the control rates.

The first order necessary conditions for this type of implicit optimal control problems have been calculated, and a direct full collocation transcription method for the numerical solution of the implicit OCP has been presented. The NLP problem resulting from the full direct collocation with a midpoint discretisation method has been accurately described, with particular attention to the sparse structure of the Jacobian and Hessian matrices. Moreover, the relationship between the NLP Lagrange multipliers and necessary conditions and the OCP ones has been derived, so as that OCP Lagrange multipliers can be estimated even if a direct method is used to solve the OCP.

Finally, the direct transcription method has been implemented in a software called *Maverick* that is currently in use at the University of Padova. This software allows to easily formulate the OCP in a symbolic algebra environment (based on *Maple*© software), then it automatically generates the OCP *C++* code and transcribes it to an NLP that is solved with third party solvers. An automatic mesh refinement algorithm is also implemented. User friendly *Python* and/or *Lua* interfaces can be used to control the solver and to specify the problem dataset. *Maverick* software has been used to solve a case-of-study implicit OCP problem, consisting in the minimum manoeuvre time of a motorbike performing a U-turn on a $3D$ road. The problem has been easily solved and the effects of the $3D$ road geometry on the motorbike dynamics have been highlighted.

form, as in the motorcycle U-turn manoeuvre example described in this chapter, and the full implicit formulation turns out to be easier to use.

Chapter 7

Conclusions

This thesis studied optimal control problems applied to vehicle minimum lap time simulations. Since OC-LTS capabilities are currently limited by the solving effectiveness of existing numerical software with high fidelity vehicle models, the intent of this work has been two-fold: first to understand which is the best approach to face this kind of problems and, second, to propose properly featured vehicle models that both improve the state-of-the of LTS models and that still lead to OCPs solvable with existing software.

The review of existing solving techniques for optimal control problems suggested that indirect and direct methods are the most suitable approaches for solving lap time simulation optimal control problems. Due to the lack of evident advantages or disadvantages of one method over the other, this work started with the comparison of indirect and direct OCP approaches when applied to OC-LTS. Even if several differences have been highlighted between the solutions provided by the chosen indirect and direct solvers, the results suggested that both methods are almost equally effective, being the observed differences due to specific numeric implementation of the representative software rather than being intrinsic to the different theoretical approaches.

This thesis then focused on the study of how different vehicle models can affect simulation outcomes. In particular it has been shown that the QSS tyre load assumption, despite being widely used in OC-LTS car models, is a too coarse simplification when accurate tyre load dynamics is required, for instance when optimising the centre of mass position or suspension design.

On the basis of this result, a novel GP2 car model for LTS has been developed. Its distinguishing features are the dynamically calculated (i.e. no QSS) tyre loads, the inclusion of the complete suspension kinematics, the ride-dependent aerodynamic forces and the full coupled tyre model. Despite the relatively high complexity of the model, the lap time simulation has been performed in a reasonable amount of time — approximately half an hour — using a common computer and an indirect approach. The model has been validated with telemetry data acquired during a qualifying lap in the 2012 GP2 season.

Go-karts have been taken into consideration in this thesis too, since in literature there is no evidence of OC-LTS applied to this peculiar type of four wheeled vehicles. A go-kart model, including with dynamic tyre loads, have been presented and validated with experimental data. The simulations showed to be able to reproduce the specific dynamics of go-karts, characterised by the under-steering behaviour in the entrance of the turns and by the lifting of the rear inner wheel.

Since electric vehicles are continuously gaining in popularity both in motor-sports and urban environment, this thesis has dealt with the optimisation of an electric vehicle too. The studied electric vehicle is a motorbike taking part at the Tourist Trophy Zero competition. Due to the length of the Snaefell Mountain Course a simple model is required in order to perform an entire simulation in a reasonable amount of time. The model presented in this thesis indeed is very concise; it captures the essence of the motorbike dynamics and focuses on the electric powertrain. This model, validated through comparison with experimental data, is able to perform a simulation on the full Snaefell Mountain Course in very few minutes. The simulations individuated the optimal motor and battery configuration for the motorbike, moreover it has been shown that the proper choice of the gear ratio can lead to a substantial increase in performance.

Finally, implicit optimal control problem have been studied. Since the implicit formulation is of great advantage from a modelling point of view, in this work a numeric software for implicit OCP has been developed. The software is based on a direct full collocation transcription method. The NLP structure resulting from the discretisation scheme adopted has been carefully presented, moreover the relationship between the OCP and NLP necessary conditions and Lagrange multipliers have been derived. The software, that has been tested and proved to work on a bench problem consisting in a minimum manoeuvre time of a motorbike on a three dimensional road, is currently in use at the University of Padova.

7.0.1 Future work

The work carried on in this thesis gives possibility for several future developments, regarding both vehicle model features for OC-LTS and numeric optimal control software.

The GP2 model can be extended to fit the current Formula 1 car architecture. While the modelling of the car dynamics (chassis, suspensions, aerodynamics, differential) for a Formula 1 car is almost the same of a GP2 one, the former is characterised by a quite intricate energy management systems which comprises a kinetic energy recovery system and a mechanism that deliver part of the engine energy either to the rear wheels (as kinetic energy) or to the internal battery storage (as electric energy). The inclusion of such feature would result in a merge between the GP2 model presented in this work and the Formula 1 model described in [58].

The tyre model used in the GP2 model may be further extended by taking into consideration for thermodynamic tyre properties and wear. It is well known that tyre characteristics highly depend on tyre temperature, thus a thermodynamic tyre model would be of great interest for more accurate lap time simulations. Moreover tyre wear is a relevant phenomenon when simulations are performed not only over a single lap but over an entire race distance. In some motor-sports categories tyre degradation is one of the major characteristics that determine the performance over the entire race, thus the knowledge of optimal tyre usage strategy is fundamental to success. These two tyre characteristics — thermodynamic behaviour and wear — have not been included in the GP2 model due to the lack both of data necessary to feed any mathematical model and of experimental data for a proper validation, but they are certainly worth to be included in a future development.

Particular attention has been dedicated to improve the tyre load transfer dynamics in OC-LTS. In all the car models used in this work, with the only exception of the go-kart one, the car chassis has been considered infinitely rigid, thus not contributing to the distribution of the tyre loads on the wheels. Whether this modelling assumption is correct or not depends highly on the car that is considered, since the chassis to suspension stiffness ratio may vary significantly from one car category to another. It would be of great interest to study the influence of the chassis compliance in determining tyre loads, and, if relevant, to include this effect in a lap time simulations. This would improve further the accuracy of lap time simulation outcomes.

Both the GP2 and go-kart models can be used for optimisation of the car design and setup. A thorough study on the optimisation of these type of vehicles with confirmation from experimental data is out of the scope of this thesis, but would be a surely interesting future work. In particular, the accurate modelling of chassis and suspension motions may give the possibility for more effective OC-LTS in the optimisation of vehicle centre of mass position and suspension design or setup.

This thesis has focused on the vehicle modelling and on the resolution of the minimum lap time optimal control problem, yet the driver model hasn't been investigated thoroughly. The driver has been assumed to be able to drive the vehicle at its maximum performance and to reproduce every driving input, provided the input has a limited maximum rate. Clearly this is a very coarse approximation of the behaviour of a real driver. Every driver has its own subjective feeling and prefers a specific car dynamical behaviour, and in motorsports it is very important to make the driver feel comfortable (in its racing meaning) with a given car. A theoretically less performing vehicle but well suited for a specific driver may allow him to achieve lower lap times. Therefore it would be useful to introduce in lap time simulations some kind of metric that gives information on how a certain driver would "like" or "dislike" driving the vehicle as the simulation suggests. The optimisation problem then would become a multi (actually, double) objective optimisation one, where the lap time and the driver's "fitness" are the two targets to minimise (the former) and maximise (the latter) together. This is certainly an interesting topic that could lead lap time simulations to take into account for human-related factors.

From the point of view of numerical OCP solution, this work presented a direct software for the transcription of implicit OCP to an NLP problem. Numerical software for OCP should be able to find the solution in a short time (compared to human scales) and should be robust. The software developed in this work offers good performances in the numerical evaluation of the NLP functions (target, Jacobian, Hessian), taking advantages of multi-core architectures. Very little work would be necessary to use GPU architecture for the evaluation of NLP functions, which can lead to a speed-up for very large problems. A significant part of the computational time is spent within the NLP solver — a third party library, like *IPOPT* — thus an improvement of this component is essential for fast OCP solutions. A possible enhancement could be the use of a bespoke linear algebra solver — used by every NLP solver — that exploits the particular structure of the Jacobian and Hessian matrices arising from the OCP nature of the NLP problem. Moreover, NLP solvers are designed to face a wide class of constrained optimisation problems; when focusing on OCP originated problems, the entire NLP solving algorithm could be tuned so as to achieve a better robustness and speed when used for this class of problems.

Bibliography

[1] M. Abramowitz and I. A. Stegun. *Handbook of mathematical functions: with formulas, graphs, and mathematical tables*, volume 55. Courier Corporation, 1964.

[2] P. Amodio and G. Romanazzi. Algorithm 859: BABDCR: a Fortran 90 package for the solution of bordered ABD linear systems. *ACM Trans. Math. Softw.*, 32(4):597–608, Dec. 2006.

[3] Z. Asus, E.-H. Aglzim, D. Chrenko, Z.-H. C. Daud, and L. Le Moyne. Dynamic modeling and driving cycle prediction for a racing series hybrid car. *IEEE Journal of Emerging and Selected Topics in Power Electronics*, 2(3):541–551, 2014.

[4] T. Back. *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms.* Oxford university press, 1996.

[5] R. Bellman. The theory of dynamic programming. Technical report, RAND Corp Santa Monica CA, 1954.

[6] C. Benini, M. Gadola, D. Chindamo, S. Uberti, F. P. Marchesin, and R. S. Barbosa. The influence of suspension components friction on race car vertical dynamics. *Vehicle System Dynamics*, 55(3):338–350, 2017.

[7] K. Berntorp, B. Olofsson, K. Lundahl, and L. Nielsen. Models and methodology for optimal trajectory generation in safety-critical road–vehicle manoeuvres. *Vehicle System Dynamics*, 52(10):1304–1332, 2014.

[8] E. Bertolazzi, F. Biral, and M. Da Lio. Symbolic-numeric efficient solution of optimal control problems for multibody systems. *Journal of computational and applied mathematics*, 185(2):404–421, 2006.

[9] E. Bertolazzi, F. Biral, and M. Da Lio. Real-time motion planning for multibody systems: Real life application examples. *Multibody System Dynamics*, 17(2–3):119–139, 2007.

[10] D. P. Bertsekas. *Nonlinear programming.* Athena scientific Belmont, 1999.

[11] J. T. Betts. *Practical methods for optimal control and estimation using nonlinear programming*, volume 19. Siam, 2010.

[12] F. Biral, E. Bertolazzi, and P. Bosetti. Notes on numerical methods for solving optimal control problems. *IEEJ Journal of Industry Applications*, 5(2):154–166, 2016.

[13] F. Biral, M. Da Lio, and E. Bertolazzi. Combining safety margins and user preferences into a driving criterion for optimal control-based computation of reference maneuvers for an adas of the next generation. In *Intelligent Vehicles Symposium, 2005. Proceedings. IEEE*, pages 36–41. IEEE, 2005.

[14] F. Biral, F. Zendri, E. Bertolazzi, P. Bosetti, M. Galvani, F. Trivellato, and M. Da Lio. A web based "virtual racing car championship" to teach vehicle dynamics and multidisciplinary design. In ASME, editor, *Proceedings of 2011 ASME International Mechanical Engineering Congress and Exposition*, pages 391–401, Denver, USA, 11th-17th November 2011. ASME, ASME.

[15] P. Bosetti, M. Da Lio, and A. Saroldi. On the human control of vehicles: an experimental study of acceleration. *European Transport Research Review*, 6(2):157–170, 2014.

[16] F. Braghin, F. Cheli, S. Melzi, and E. Sabbioni. Race driver model. *Comput. Struct.*, 86(13-14):1503–1516, July 2008.

[17] D. Brayshaw. *Use of numerical optimisation to determine on-limit handling behaviour of race cars.* PhD thesis, Cranfield University, 2004.

[18] D. Brayshaw and M. Harrison. A quasi steady state approach to race car lap simulation in order to understand the effects of racing line and centre of gravity location. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 219(6):725–739, 2005.

[19] D. Brayshaw and M. Harrison. Use of numerical optimization to determine the effect of the roll stiffness distribution on race car performance. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 219(10):1141–1151, 2005.

[20] A. E. Bryson. *Dynamic optimization*, volume 1. Prentice Hall, 1999.

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

[22] D. Casanova, R. S. Sharp, and P. Symonds. Minimum time manoeuvring: The significance of yaw inertia. *Vehicle system dynamics*, 34(2):77–115, 2000.

[23] D. Casanova, R. S. Sharp, and P. Symonds. On the optimisation of the longitudinal location of the mass centre of a formula one car for two circuits. In *Proceedings of AVEC*, volume 2, pages 6–12, 2002.

[24] D. Cole, A. Pick, and A. Odhams. Predictive and linear quadratic methods for potential application to modelling driver steering control. *Vehicle System Dynamics*, 44(3):259–284, 2006.

[25] V. Cossalter, M. Da Lio, R. Lot, and L. Fabbri. A general method for the evaluation of vehicle manoeuvrability with special emphasis on motorcycles. *Vehicle system dynamics*, 31(2):113–135, 1999.

[26] V. Cossalter, A. Doria, R. Lot, N. Ruffo, and M. Salvador. Dynamic properties of motorcycle and scooter tires: measurement and comparison. *Vehicle system dynamics*, 39(5):329–352, 2003.

[27] V. Cossalter, R. Lot, and D. Tavernini. Optimization of the centre of mass position of a racing motorcycle in dry and wet track by means of the "optimal maneuver method". In *Mechatronics (ICM), 2013 IEEE International Conference on*, pages 412–417. IEEE, 2013.

[28] V. Cossalter, M. Peretto, and S. Bobbo. Investigation of the influences of tyre—road friction and engine power on motorcycle racing performance by means of the optimal manoeuvre method. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 224(4):503–519, 2010.

[29] D. Crolla, Q. Ren, S. ElDemerdash, and F. Yu. Controller design for hybrid vehicles - state of the art review. 2008. cited By 18.

[30] M. Da Lio. Analisi della manovrabilita dei veicoli: un approccio basato sul controllo ottimo. *ATA-TORINO-*, 50:35–42, 1997.

[31] F. De Groote, A. Kinney, A. Rao, and B. Fregly. Evaluation of direct collocation optimal control problem formulations for solving the muscle redundancy problem. *Annals of Biomedical Engineering*, 44(10):2922–2936, 2016. cited By 3.

[32] K. Deb. Multi-objective optimisation using evolutionary algorithms: an introduction. In *Multi-objective evolutionary optimisation for product design and manufacturing*, pages 3–34. Springer, 2011.

[33] M. Diehl, D. B. Leineweber, and A. Schäfer. *MUSCOD-II Users' Manual*. Universität Heidelberg, Heidelberg, iwr-preprint 2001-25 edition, 2001.

[34] I. S. Duff. MA57–a code for the solution of sparse symmetric definite and indefinite systems. *ACM Trans. Math. Softw.*, 30(2):118–144, June 2004.

[35] P. Falugi, E. Kerrigan, and E. van Wyk. *Imperial college london optimal control software user guide (ICLOCS)*. Imperial College London, London, May 2010.

[36] M. Gadola, D. Vetturi, D. Cambiaghi, and L. Manzo. A tool for lap time simulation. Technical report, SAE Technical Paper, 1996.

[37] M. Gerdts. A moving horizon technique for the simulation of automobile test-drives. *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik*, 83(3):147–162, 2003.

[38] M. Gerdts. A survey on optimal control problems with differential-algebraic equations. In A. Ilchmann and T. Reis, editors, *Surveys in Differential-Algebraic Equations II*, pages 103–161. Springer International Publishing, Cham, 2015.

[39] M. Gerdts. A survey on numerical methods for the simulation of initial value problems with sdaes. In A. Ilchmann and T. Reis, editors, *Surveys in Differential-Algebraic Equations IV*, pages 221–300. Springer International Publishing, Cham, 2017.

[40] M. Gerdts, S. Karrenberg, B. Müller-Beßler, and G. Stock. Generating locally optimal trajectories for an automatically driven car. *Optimization and Engineering*, 10(4):439, 2008.

[41] S. B. Gershwin and D. H. Jacobson. A discrete-time differential dynamic programming algorithm with application to optimal orbit transfer. *AIAA Journal*, 8(9):1616–1626, 2017/08/29 1970.

[42] T. D. Gillespie. Fundamentals of vehicle dynamics. Technical report, SAE Technical Paper, 1992.

[43] L. Guzzella, A. Sciarretta, et al. *Vehicle propulsion systems*, volume 1. Springer, 2007.

[44] J. Hendrikx, T. Meijlink, and R. Kriens. Application of optimal control theory to inverse simulation of car handling. *Vehicle System Dynamics*, 26(6):449–461, 1996.

[45] A. Hughes and B. Drury. *Electric motors and drives: fundamentals, types and applications*. Newnes, 2013.

[46] F. John. Extremum problems with inequalities as side constraints. in “studies and essays, courant anniversary volume”(ko friedrichs, oe neugebauer and jj stoker, eds), 187-204, 1948.

[47] S. D. Keen and D. J. Cole. Steering control using model predictive control and multiple internal models. In *Proceedings of the 8th International Symposium on Automotive Control (AVEC 2006)*, pages 599–604. Citeseer, 2006.

[48] D. P. Kelly. *Lap time simulation with transient vehicle and tyre dynamics*. PhD thesis, Cranfield University, 2008.

[49] D. P. Kelly and R. S. Sharp. Time-optimal control of the race car: influence of a thermodynamic tyre model. *Vehicle System Dynamics*, 50(4):641–662, 2012.

[50] C. Kirches, S. Sager, H. G. Bock, and J. P. Schlöder. Time-optimal control of automobile test drives with gear shifts. *Optimal Control Applications and Methods*, 31(2):137–153, 2010.

[51] S. Koehler, A. Viehl, O. Bringmann, and W. Rosenstiel. Improved energy efficiency and vehicle dynamics for battery electric vehicles through torque vectoring control. volume 2015-August, pages 749–754, 2015.

[52] C. Lanczos. *The variational principles of mechanics*. Courier Corporation, 2012.

[53] G. Lantoine and R. P. Russell. A hybrid differential dynamic programming algorithm for constrained optimal control problems. part 1: Theory. *Journal of Optimization Theory and Applications*, 154(2):382–417, 2012.

[54] G. Lantoine and R. P. Russell. A hybrid differential dynamic programming algorithm for constrained optimal control problems. part 2: Application. *Journal of Optimization Theory and Applications*, 154(2):418–442, Aug 2012.

[55] D. Limebeer. Optimising the aero-suspension interactions in a forumla one car. *IEEE Transactions on Control Systems Technology*, 8 2014.

[56] D. Limebeer and G. Perantoni. Optimal control of a formula one car on a three-dimensional track—part 2: Optimal control. *Journal of Dynamic Systems, Measurement, and Control*, 137(5):051019, 2015.

[57] D. J. Limebeer and M. Massaro. *Dynamics and optimal control of road vehicles.* Oxford University Press, New York, in press.

[58] D. J. Limebeer, G. Perantoni, and A. Rao. Optimal control of formula one car energy recovery systems. *International Journal of Control*, 87(10):2065–2080, 2014.

[59] R. Lot and F. Biral. A curvilinear abscissa approach for the lap time optimization of racing vehicles. volume 47, pages 7559–7565. Elsevier, 2014.

[60] R. Lot and M. Da Lio. A symbolic approach for automatic generation of the equations of motion of multibody systems. *Multibody System Dynamics*, 12(2):147–172, 2004.

[61] R. Lot and S. Evangelou. Lap time optimization of a sports series hybrid electric vehicle. In *2013 World Congress on Engineering*, 2013.

[62] M. Maniowski. Optimisation of driver actions in rwd race car including tyre thermodynamics. *Vehicle System Dynamics*, 54(4):526–544, 2016.

[63] M. Maniowski. Optimization of driver and chassis of fwd racing car for faster cornering. In *The Dynamics of Vehicles on Roads and Tracks: Proceedings of the 24th Symposium of the International Association for Vehicle System Dynamics (IAVSD 2015), Graz, Austria, 17-21 August 2015*, page 77. CRC Press, 2016.

[64] F. Marchesin, R. Barbosa, M. Alves, M. Gadola, D. Chindamo, and C. Benini. Upright mounted pushrod: The effects on racecar handling dynamics. In *Proceedings of the 24th Symposium of the International Association for Vehicle System Dynamics, IAVSD, Graz*, 2015.

[65] M. I. Masouleh and D. J. Limebeer. Optimizing the aero-suspension interactions in a formula one car. *IEEE Transactions on Control Systems Technology*, 24(3):912–927, 2016.

[66] D. Mayne. A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems. *International Journal of Control*, 3(1):85–95, 1966.

[67] L. Meirovitch. *Methods of analytical dynamics.* Courier Corporation, 2010.

[68] D. Metz and D. Williams. Near time-optimal control of racing vehicles. *Automatica*, 25(6):841–857, 1989.

[69] G. Mirone. Multi-body elastic simulation of a go-kart: Correlation between frame stiffness and dynamic performance. *International Journal of Automotive Technology*, 11(4):461–469, 2010.

[70] H. Pacejka. *Tire and vehicle dynamics.* Elsevier, 2006.

[71] M. A. Patterson and A. V. Rao. Gpops-ii: A matlab software for solving multiple-phase optimal control problems using hp-adaptive gaussian quadrature collocation methods and sparse nonlinear programming. *ACM Transactions on Mathematical Software (TOMS)*, 41(1):1, 2014.

[72] G. Perantoni and D. J. Limebeer. Optimal control for a formula one car with variable parameters. *Vehicle System Dynamics*, 52(5):653–678, 2014.

[73] G. Perantoni and D. J. Limebeer. Optimal control of a formula one car on a three-dimensional track—part 1: Track modeling and identification. *Journal of Dynamic Systems, Measurement, and Control*, 137(5):051018, 2015.

[74] L. V. Pérez and E. A. Pilotta. Optimal power split in a hybrid electric vehicle using direct transcription of an optimal control problem. *Mathematics and Computers in Simulation*, 79(6):1959–1970, 2009.

[75] E. R. Pinch. *Optimal control and the calculus of variations.* Oxford University Press Oxford, 1993.

[76] L. S. Pontryagin. *Mathematical theory of optimal processes.* John Wiley & Sons, 1962.

[77] A. Quarteroni, R. Sacco, and F. Saleri. *Matematica numerica.* Springer Science & Business Media, 2008.

[78] A. V. Rao. A survey of numerical methods for optimal control. *Advances in the Astronautical Sciences*, 135(1):497–528, 2009.

[79] L. Rodgers, R. Gogoana, T. German, M. Jeunnette, R. Briggs, E. Fuentes, and W. Pritchett. Designing an electric motorcycle for the isle of man tt zero race, and how electric vehicle racing could be used to spur innovation. In *EVS26 International Battery, Hybrid and Fuel Cell Electric Vehicle Symposium*, 2012.

[80] F. R. Salmasi. Control strategies for hybrid electric vehicles: Evolution, classification, comparison, and future trends. *IEEE Transactions on vehicular technology*, 56(5):2393–2404, 2007.

[81] S. M. Savaresi, C. Spelta, D. Ciotti, M. Sofia, E. Rosignoli, and E. Bina. Virtual selection of the optimal gear-set in a race car. *International Journal of Vehicle Systems Modelling and Testing*, 3(1-2):47–67, 2008.

[82] H. Scherenberg. Mercedes-benz racing design and cars experience. Technical report, SAE Technical Paper, 1958.

[83] R. Sharp. A method for predicting minimum-time capability of a motorcycle on a racing circuit. *Journal of Dynamic Systems, Measurement, and Control*, 136(4):041007, 2014.

[84] B. Siegler, A. Deakin, and D. Crolla. Lap time simulation: Comparison of steady state, quasi-static and transient recing car cornering stategies. In S. of Automotive Engineers, editor, *2000 SAE Motorsports Engineering conference amd Exposition*, number paper 2000–01-3563, page 9. SAE International, 2000.

[85] B. Simon, C. Vittore, M. Matteo, and P. Martino. Application of the "optimal maneuver method" for enhancing racing motorcycle performance. *SAE International Journal of Passenger Cars-Mechanical Systems*, 1(2008-01-2965):1311–1318, 2008.

[86] A. Sorniotti and M. Curto. Racing simulation of a formula 1 vehicle with kinetic energy recovery system. In *SAE Technical Paper*, page 15. SAE International, 12 2008.

[87] S. Stockar, V. Marano, M. Canova, G. Rizzoni, and L. Guzzella. Energy-optimal control of plug-in hybrid electric vehicles for real-world driving cycles. *IEEE Transactions on Vehicular Technology*, 60(7):2949–2962, 2011.

[88] J. Stoer and R. Bulirsch. *Introduction to numerical analysis*, volume 12. Springer Science & Business Media, 2013.

[89] M. Tanelli, M. Corno, and S. Saveresi. *Modelling, Simulation and Control of Two-wheeled Vehicles*. John Wiley & Sons, 2014.

[90] D. Tavernini, M. Massaro, E. Velenis, D. I. Katzourakis, and R. Lot. Minimum time cornering: the effect of road surface and car transmission layout. *Vehicle System Dynamics*, 51(10):1533–1547, 2013.

[91] D. Tavernini, E. Velenis, R. Lot, and M. Massaro. The optimality of the handbrake cornering technique. *Journal of Dynamic Systems, Measurement, and Control*, 136(4):041019, 2014.

[92] A. Tremlett and D. Limebeer. Optimal tyre usage for a formula one car. *Vehicle System Dynamics*, 54(10):1448–1473, 2016.

[93] P. Viviani and T. Flash. Minimum-jerk, two-thirds power law, and isochrony: converging approaches to movement planning. *Journal of Experimental Psychology: Human Perception and Performance*, 21(1):32, 1995.

[94] T. Völkl, M. Muehlmeier, and H. Winner. Extended steady state lap time simulation for analyzing transient vehicle behavior. *SAE International Journal of Passenger Cars-Mechanical Systems*, pages 283–292, 2013.

[95] A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1):25–57, 2006.

[96] J. Weigl, M. Henz, Inayati, and H. Saidi. Converted battery-powered electric motorcycle and hydrogen fuel cell-powered electric motorcycle in south east asia: Development and performance test. pages 1–4, 2016.

[97] J. Weigl, H. Saidi, and Inayati. Design, testing and optimisation of a hydrogen fuel cell motorcycle for south east asia. 2013. cited By 1.

[98] M. J. Weinstein and A. V. Rao. Algorithm 984: Adigator, a toolbox for the algorithmic differentiation of mathematical functions in matlab using source transformation via operator overloading. *ACM Trans. Math. Softw.*, 44(2):21:1–21:25, Aug. 2017.

[99] S. Wright and J. Nocedal. Numerical optimization. *Springer Science*, 35:67–68, 1999.

Appendices

Appendix A

Road model

Optimal control simulations of road vehicles always rely upon a road model that allows to track the position and orientation of the vehicle with respect to the road and to the ground. In this thesis the road is modelled as a ribbon characterised by a certain orientation and width that both vary along the road. This model captures the road slope and banking, which are the main features of a three dimensional road; irregularities and bumps are not considered, yet they could be added as a further work.

The road ribbon $\boldsymbol{R}$, that is a bi-dimensional surface within the three-dimensional space, can be described by means of its middle lane $\mathcal{C}(s)$, its orientation $\boldsymbol{O}_r$ with respect to the ground frame $\boldsymbol{W}_g$ (assumed fixed) and its width r_w. The road middle lane $\mathcal{C}(s)$ is a 1-dimensional parametric curve function of the independent parameter s that corresponds to the distance travelled along the road middle lane:

$$\mathcal{C}(s) = \Big[x_r(s); \quad y_r(s); \quad z_r(s)\Big] \tag{A.1}$$

where x_r,y_r and z_r are the Cartesian coordinates of the road middle lane with respect to the ground frame.

In order to describe the ribbon orientation matrix $\boldsymbol{O}_r(s)$, a ribbon frame has to be defined first. The ribbon frame is here defined as an orthonormal frame centred in $\mathcal{C}(s)$, with the x-axis unit vector $\boldsymbol{e}_x$ aligned with the road middle lane, the y-axis unit vector $\boldsymbol{e}_y$ perpendicular to the e_x vector and lying on a line that intersect the road right border, and the z-axis unit vector $\boldsymbol{e}_z$ given by the inner product between $\boldsymbol{e}_x$ and $\boldsymbol{e}_y$, so as it is perpendicular to the road surface. Mathematically, the unit vectors that determines the ribbon orientation matrix are defined as follows:

$$\begin{aligned} \boldsymbol{e}_x(s)\left\|\frac{d\mathcal{C}(s)}{ds}\right\| &= \frac{d\mathcal{C}(s)}{ds} \\ \boldsymbol{e}_y(s) &= \boldsymbol{v} \quad : \quad \boldsymbol{v}\perp\boldsymbol{e}_x \ \wedge\|\boldsymbol{v}\| = 1 \ \wedge \ \exists k \neq 0 : k\boldsymbol{v} \in \boldsymbol{b}_r(s) \\ \boldsymbol{e}_z(s) &= \boldsymbol{e}_x(s) \times \boldsymbol{e}_y(s) \end{aligned} \tag{A.2}$$

where $\boldsymbol{b}_r(s)$ is the road right border parametric curve and $\wedge$ is the *and* operator. It should be noted that SAE convention is used for the axis orientation. The ribbon frame orientation matrix $\boldsymbol{O}_r(s)$ (w.r.t the ground) is the matrix whose columns contain respectively the coordinates of the unit vectors $\boldsymbol{e}_x$, $\boldsymbol{e}_y$, $\boldsymbol{e}_z$ expressed in the ground frame. The ribbon frame orientation matrix $\boldsymbol{O}_r(s)$ can be fully described by three parameters that are related to the three different rotations that allow to rotate from the ground frame to the road ribbon frame. Many conventions can be used to express these three parameters [67]. Here the zyx convention is adopted, i.e. the three parameters correspond to three rotations, the first about the z-axis, the second about the y-axis and the third about the x one:

$$\begin{aligned} \boldsymbol{O}_r(s) &= R_z(\psi)R_y(\sigma)R_x(\beta) \\ &= \begin{bmatrix} \cos\psi\cos\sigma & -\sin\psi\cos\beta+\cos\psi\sin\sigma\sin\beta & \sin\psi\sin\beta+\cos\psi\sin\sigma\cos\beta \\ \sin\psi\cos\sigma & \cos\psi\cos\beta+\sin\psi\sin\sigma\sin\beta & -\cos\psi\sin\beta+\sin\psi\sin\sigma\cos\beta \\ -\sin\sigma & \cos\sigma\sin\beta & \cos\sigma\cos\beta \end{bmatrix} \end{aligned} \tag{A.3}$$

where $R_i(\xi)$ is the rotation of an angle ξ about the $i-th$ Cartesian axis (x, y or z). The angles ψ, σ and β are respectively the road heading (i.e. the direction of travelling), slope (i.e. travelling uphill or downhill) and banking (i.e. the road leaning). In the case of a flat road the slope σ and banking β angles are always zero. With the definitions given above, it follows that the road middle lane $\mathcal{C}(s)$ can be obtained by integrating in s the first column of the road ribbon frame $\boldsymbol{O}_r(s)$:

$$\mathcal{C}(s) = \begin{bmatrix} x_r(s) \\ y_r(s) \\ z_r(s) \end{bmatrix} = \int_0^s \begin{bmatrix} \cos\psi(t)\cos\sigma(t) \\ \sin\psi(t)\cos\sigma(t) \\ -\sin\sigma(t) \end{bmatrix} dt \tag{A.4}$$

The road middle lane $\mathcal{C}(s)$ and orientation matrix $\boldsymbol{O}_s(s)$ can be put together into the 4×4 road frame matrix $\boldsymbol{W}_r$ that fully determines the road position and orientation:

$$\boldsymbol{W}_r(s) = \begin{bmatrix} \cos\psi\cos\sigma & -\sin\psi\cos\beta+\cos\psi\sin\sigma\sin\beta & \sin\psi\sin\beta+\cos\psi\sin\sigma\cos\beta & x_r \\ \sin\psi\cos\sigma & \cos\psi\cos\beta+\sin\psi\sin\sigma\sin\beta & -\cos\psi\sin\beta+\sin\psi\sin\sigma\cos\beta & y_r \\ -\sin\sigma & \cos\sigma\sin\beta & \cos\sigma\cos\beta & z_r \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.5}$$

The road frame $\boldsymbol{W}_r$ is shown in figure A.1. The scalar ribbon width $r_w(s)$ completes the road characterisation.

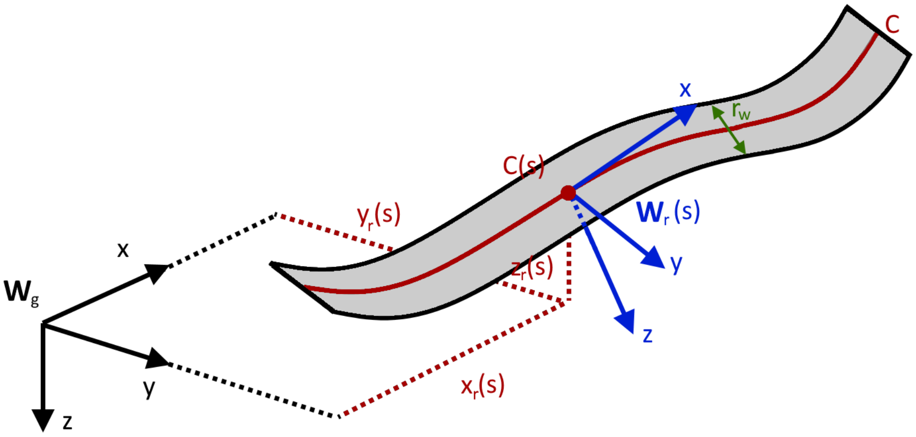


Figure A.1: Coordinate system of the road model. The road frame $\boldsymbol{W}_r$ with respect to the ground frame $\boldsymbol{W}_g$ and the road width are shown.

Vehicle tracking

When a vehicle moves on the road, its reference frame $\boldsymbol{W}_v(t)$ can be expressed relatively to the road frame $\boldsymbol{W}_r(s(t))$:

$$\boldsymbol{W}_v(t) = \boldsymbol{W}_r(s(t))\boldsymbol{W}_{rel}(\boldsymbol{x}_{rel}(t)) \tag{A.6}$$

where $s(t)$ is the position of the vehicle along the road middle lane (now seen as a function of time) and $\boldsymbol{W}_{rel}$ is the relative frame matrix of the vehicle with respect to the road frame that depends on the vehicle degrees of freedom $\boldsymbol{x}_{rel}(t)$ relative to the road. Depending on the vehicle model, $\boldsymbol{x}_{rel}$ can comprise up to five dof: indeed a rigid body may have up to six dof, but one of the six is the curvilinear coordinate $s(t)$.

When calculating the vehicle equations of motion, velocities of the road frame $\boldsymbol{W}_r(s(t))$ generate apparent forces that must be taken into account. The translational $\boldsymbol{v}_v$ and angular $\boldsymbol{\omega}_v$ velocities of the vehicle frame can be expressed as the sum of the road frame and relative frame velocities:

$$\boldsymbol{v}_v = \boldsymbol{v}_r + \boldsymbol{v}_{rel}, \quad \boldsymbol{\omega}_v = \boldsymbol{\omega}_r + \boldsymbol{\omega}_{rel} \tag{A.7}$$

where $\boldsymbol{v}_r$, $\boldsymbol{\omega}_r$ are respectively the translational and angular velocities of the road frame and $\boldsymbol{v}_{rel}$, $\boldsymbol{\omega}_{rel}$ are the velocities of the frame $\boldsymbol{W}_{rel}$ with respect to the road frame $\boldsymbol{W}_r$. If Newton and Euler equations for the whole vehicle are calculated within the vehicle frame $\boldsymbol{W}_{rel}$, they assume a very simple form where the net external forces $\boldsymbol{F}$ and torques $\boldsymbol{\tau}$ are equated to respectively to the inertial forces $\boldsymbol{F}_i$ and inertial torques $\boldsymbol{\tau}_i$:

$$\begin{aligned} \boldsymbol{F} &= \boldsymbol{F}_i \\ \boldsymbol{\tau} &= \boldsymbol{\tau}_i \end{aligned} \tag{A.8}$$

While the external forces and torques at the left hand side of the above equations depend on the vehicle model, the inertial forces and torques at the right hand side of the above equations are equal to:

$$\begin{aligned} \boldsymbol{F}_i &= m_v(\dot{\boldsymbol{v}}_v + \boldsymbol{\omega}_v \times \boldsymbol{v}_v) \\ \boldsymbol{\tau}_i &= \boldsymbol{I}_v\dot{\boldsymbol{\omega}}_v + \boldsymbol{\omega}_v \times \boldsymbol{I}\boldsymbol{\omega}_v \end{aligned} \tag{A.9}$$

where m_v is the total mass and $\boldsymbol{I}_v$ is the total inertia tensor.

The relative frame velocities $\boldsymbol{v}_{rel}$, $\boldsymbol{\omega}_{rel}$ depend only on the time derivative of the relative degree of freedom $\dot{\boldsymbol{x}}_{rel}(t)$, moreover the road velocities $\boldsymbol{v}_v$, $\boldsymbol{\omega}_v$ depend only on the time derivative of the curvilinear abscissa $\dot{s}(t)$ and on the road characteristics. The translational speed $\boldsymbol{v}_r$ of the road frame is given by the following simple expression, being only the x component nonzero:

$$\boldsymbol{v}_r(\dot{s}(t)) = \dot{s}(t)\left[1; \quad 0; \quad 0\right] \tag{A.10}$$

The angular velocity of the road frame too has a simple expression if it is expressed in terms of the road curvatures $\kappa(s)$, $\nu(s)$, $\tau(s)$: κ is curvature in the $x - y$ transversal plane, ν is the curvature in the sagittal plane $x - z$ and τ is the torsion:

$$\boldsymbol{\omega}_r(\dot{s}(t)) = \dot{s}(t)\left[\tau(s(t)); \quad \nu(s(t)); \quad \kappa(s(t))\right] \tag{A.11}$$

The road curvatures can be expressed as function of the road frame angle rates:

$$\begin{aligned} \kappa &= \cos\sigma\cos\beta\psi' - \sin\beta\sigma' \\ \nu &= \cos\sigma\sin\beta\psi' + \cos\beta\sigma' \\ \tau &= \beta' - \sin\sigma\psi' \end{aligned} \tag{A.12}$$

where the superscript $'$ indicates the derivative with respect to the curvilinear abscissa s. In the case of a flat road the only curvature that is non-zero is κ.

It has been shown that the road geometry generate inertial forces that appear in the Newton and Euler equations of motion. Even if Newton and Euler equations can be written in a different form than (A.9), inertial forces due to the road must appear. Moreover, road curvatures defined in (A.12) allows to easily express the road frame angular velocities as function of $\dot{s}$ only.

Time to space domain transformation

In optimal control lap time simulations it is more convenient to use the road curvilinear abscissa s as independent variable rather than the time t, since practical experience shows that problems so formulated are easier to be solved by optimal control numerical solvers [25, 59, 57]. In particular, the curvilinear abscissa s can be used as independent variable if and only if the function $s(t)$ is invertible, so as that a bijective map from t to s exists. This is in general true if the vehicle never inverts its motion and if the road width is always less than the road local curvature radius, in which case the road surface could overlap itself and the function $s(t)$ may be non injective.

When a bijective map from t to s exists, the curvilinear abscissa s can be removed from the state variables of the optimal control problem and can be made the independent variable. Denoting by $\boldsymbol{x}$ is the state variable vector of the time-domain problem and by $\boldsymbol{x}_s$ is the state variable vector for the space domain problem, the following relationship holds:

$$\boldsymbol{x} = \begin{bmatrix} s; & \boldsymbol{x}_s \end{bmatrix} \tag{A.13}$$

Once s becomes the independent variable, all the derivatives w.r.t the time that appear in the equations of motion must be expressed as derivative w.r.t s. This can be easily done using the chain derivative rule:

$$\frac{d\boldsymbol{x}_s(t)}{dt} = \frac{d\boldsymbol{x}_s(s)}{ds}\frac{ds(t)}{dt} = \frac{d\boldsymbol{x}_s(s)}{ds}\dot{s} \tag{A.14}$$

Using the above expression, all the functions $\boldsymbol{f}_t\left(\boldsymbol{x}_s(t), \frac{d\boldsymbol{x}_s(t)}{dt}\right)$ in the original time domain optimal control problem can be expressed as function of the space-derivatives $\boldsymbol{f}_s\left(\boldsymbol{x}_s(s), \frac{d\boldsymbol{x}_s(s)}{ds}\right)$ in the space domain optimal control problem:

$$\boldsymbol{f}_s\left(\boldsymbol{x}_s(s(t)), \frac{d\boldsymbol{x}_s(s(t))}{ds}\right) = \boldsymbol{f}_t\left(\boldsymbol{x}_s(t), \frac{d\boldsymbol{x}_s(t)}{dt}\frac{1}{\dot{s}}\right) \tag{A.15}$$

Looking at the above transformation, it is clear that the time derivative of s cannot be zero, otherwise the right hand side would be undefined. However, under the hypothesis that there exists a bijective map from t to s, $\dot{s}$ is never zero otherwise the map would not be bijective.

Finally, it should be noted that the quantity $\dot{s}$ must be expressed as a function of the new state variables $\boldsymbol{x}_s$. This can be achieved finding a kinematic relationship that expresses the longitudinal speed of the road frame (that is exactly $\dot{s}$, see equation (A.10)) as a function of the other velocities. In certain cases this expression may be not trivial, and since all the time-derivatives in the first order equations are multiplied by $1/\dot{s}$ (see equation (A.15)), the time to space domain transformation may lead to a substantial increase in the complexity of the equations. This problem can be avoided by keeping $\dot{s}$ as an algebraic state variable in the space domain optimal control problem.

Appendix B

Road data reconstruction

The road model presented in appendix A fully describes the road geometry by means of the road middle lane $\mathcal{C}(s)$, width r_w and orientation angles (heading ψ, slope σ and banking β), or equivalently the curvatures (κ, ν, τ), as function of the road curvilinear abscissa s. It is thus necessary to know such data in order to perform any simulation on the given road. From a practical point of view, the road data that can be measured are the coordinates of a set of points belonging to the road left border $\left[x_{rl0}^{(i)};\ y_{rl0}^{(i)};\ z_{rl0}^{(i)}\right], i = 1, \ldots, N$ and the set of corresponding right border point coordinates[1] $\left[x_{rr0}^{(i)};\ y_{rr0}^{(i)};\ z_{rr0}^{(i)}\right], i = 1, \ldots, N$. These sets of coordinates can be obtained, for instance, with GPS sensors or from *Google* satellite images. In the latter case, satellite images give a flat representation of the road (i.e. the elevation is not considered), thus they must combined with another tool that provides the missing elevation data (e.g. `www.gpsvisualizer.com`). When the sets of points are obtained, the position of the road middle lane can be estimated. Since the road middle lane is equidistant (by definition) from the road borders, the coordinates of N points belonging to the middle lane $\left[x_{r0}^{(i)}; y_{r0}^{(i)}; z_{r0}^{(i)}\right], i = 1, \ldots, N$ can be calculated simply taking the average of the coordinates on the right and left borders:

$$\left[x_{r0}^{(i)};\quad y_{r0}^{(i)};\quad z_{r0}^{(i)}\right] = \frac{\left[x_{rl0}^{(i)};\quad y_{rl0}^{(i)};\quad z_{rl0}^{(i)}\right] + \left[x_{rr0}^{(i)};\quad y_{rr0}^{(i)};\quad z_{rr0}^{(i)}\right]}{2} \qquad i = 1, \ldots, N \tag{B.1}$$

The sets of middle lane points can be interpolated by splines so as to obtain an estimate of the middle lane $1D$-parametric curve $\boldsymbol{s}_0$, moreover the arc length of this curve is an estimate of the curvilinear abscissa s_0. The right and left border point coordinates can similarly be

[1] It is here assumed that the n-th left border point corresponds to the n-th right border point, i.e. the line connecting these two points is perpendicular to the road middle lane. If the sets of points do not satisfy this property, some algorithms has to be used in order to obtain two sets of points satisfying such property. The description of such algorithm, that depends on the methodology used to extract the point coordinates, is out of the scope of this thesis.

interpolated to obtain an estimate of the coordinates as function of the estimated curvilinear abscissa s_0:

$$\begin{aligned} \boldsymbol{s}_0(s_0) &= \Big[x_{r0}(s_0); \quad y_{r0}(s_0); \quad z_{r0}(s_0)\Big] \\ \boldsymbol{b}_{l0}(s_0) &= \Big[x_{rl0}(s_0); \quad y_{rl0}(s_0); \quad z_{rl0}(s_0)\Big] \\ \boldsymbol{b}_{r0}(s) &= \Big[x_{rr0}(s_0); \quad y_{rr0}(s_0); \quad z_{rr0}(s_0)\Big] \end{aligned} \tag{B.2}$$

where $\boldsymbol{b}_{r0}(s_0)$ and $\boldsymbol{b}_{l0}(s_0)$ are the estimates of the road right and left borders.

Once the estimates of the road right and left borders are obtained, the road heading ψ and slope σ angles could be calculated differentiating equation (A.4). Moreover the banking angle β and road width r_w could be obtained by imposing the road borders to pass through the border estimates $\boldsymbol{b}_{r0}(s_0)$, $\boldsymbol{b}_{l0}(s_0)$. However, the road data $\boldsymbol{b}_{r0}(s_0)$, $\boldsymbol{b}_{l0}(s_0)$ is affected by a certain noise due to the measurement process, and the differentiation of equation (A.4) would amplify it. The noise can be reduced by calculating the road data not directly, as described just above, but through an optimal control problem where some regularisation terms are included to ensure a certain smoothness of the resulting road. The procedure is very similar to the one described in [73], and it is here reported.

The optimal control problem consists in finding the road geometry (i.e. $\alpha(s)$, $\sigma(s)$, $\beta(s)$, $r_w(s)$) that minimises the distance of the road borders $\boldsymbol{b}_r$, $\boldsymbol{b}_l$ from their estimates $\boldsymbol{b}_{r0}$, $\boldsymbol{b}_{l0}$, and ensuring at the same time a certain smoothness. Mathematically, the optimal control problem can be formulated as follows:

$$\min_{\boldsymbol{u}} \quad \int_0^{s_{0f}} S(\boldsymbol{x}) + w_u U(\boldsymbol{u}) \quad ds_0 \tag{B.3a}$$

subject to:

$$\boldsymbol{x}' = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \tag{B.3b}$$

where $\boldsymbol{x}$ is the state variables vector that contains the road geometry variables, $\boldsymbol{u}$ is the control vector and $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})$ are the first order equations.

The state variables vector includes the road middle lane coordinates, the road orientation angles, the road width and the road curvilinear abscissa s: $\boldsymbol{x} = [x_r;\ y_r;\ z_r;\ \alpha;\ \sigma;\ \beta;\ r_w;\ s]$. The variable s is a state of the optimal control problem because the road curvilinear abscissa is not known before solving the OCP, and only its estimate, s_0 is available. The independent variable of the OCP is the road curvilinear abscissa estimate (and s_{0f} is its value at the end of the road).

The control vector $\boldsymbol{u}$ determines the variation of the state variables with s_0; thus it includes u_{rw} that controls the road width rate r_w', u_s that controls the curvilinear abscissa rate s' and three variables u_1, u_2, u_3 that control the first order derivatives of the road angles α', σ', β'. In particular the controls u_1, u_2, u_3 can be either the exact derivatives of the road angles, or the road curvatures κ, ν, τ; indeed the road angle rates and the curvatures express the same information and are related through equation (A.12). Here the controls u_1, u_2, u_3 are assumed to be road angle rates.

Summarising, the state and control vector are the following:

$$\begin{aligned} \boldsymbol{x} &= \Big[x_r; \quad y_r; \quad z_r; \quad \psi; \quad \sigma; \quad \beta; \quad r_w; \quad s\Big] \\ \boldsymbol{u} &= \Big[u_{rw}; \quad u_s; \quad u_\psi; \quad u_\sigma; \quad u_\beta\Big] \end{aligned} \tag{B.4}$$

and the first order equations $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})$ are:

$$\boldsymbol{x}' = \begin{bmatrix} x_r' \\ y_r' \\ z_r' \\ \psi' \\ \sigma' \\ \beta' \\ r_w' \\ s' \end{bmatrix} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) = \begin{bmatrix} u_s\left(\cos\psi\cos\sigma\right) \\ u_s\left(\sin\psi\cos\sigma\right) \\ -u_s\sin\sigma \\ u_\psi \\ u_\sigma \\ u_\beta \\ u_{rw} \\ u_s \end{bmatrix} \tag{B.5}$$

It can be noticed that in the first three entries of $\boldsymbol{f}$ the term u_s (that is equal to $\frac{ds}{ds_0}$) makes the derivatives of the road angles to be expressed w.r.t s_0 and not s. If the estimate of the road curvilinear abscissa s_0 is close to the real one s, then u_s is approximately equal to one. If the road curvatures κ, ν, τ would have been chosen in place of the controls u_ψ, u_σ, u_β, then the fourth to sixth equations in $\boldsymbol{f}$ would have been different.

The OCP target in (B.3) is composed by two terms: the first, $S(\boldsymbol{x})$, is related to the distance of the road borders from the estimated ones, and the second, $U(\boldsymbol{u}')$, is the regularisation term (w_u is a constant):

$$S(\boldsymbol{x}) = \|\boldsymbol{b}_r - \boldsymbol{b}_{r0}\|^2 + \|\boldsymbol{b}_l - \boldsymbol{b}_{l0}\|^2 \tag{B.6a}$$

$$U(\boldsymbol{u}') = \left\| \left[\frac{u_{rw}}{\sigma_{u_{rw}}} + \frac{u_s}{\sigma_{u_s}} + \frac{u_\psi}{\sigma_{u_\psi}} + \frac{u_\sigma}{\sigma_{u_\sigma}} + \frac{u_\beta}{\sigma_{u_\beta}} \right] \right\|^2 \tag{B.6b}$$

where $\boldsymbol{b}_r$, $\boldsymbol{b}_l$ are the coordinates of the road right and left borders, $\boldsymbol{b}_{r0}$, $\boldsymbol{b}_{l0}$ are their estimates and σ_{u_s}, $\sigma_{u_{rw}}$, u_ψ, u_σ, u_β are scaling coefficients. It can be noted that the term U penalises the control derivatives so as to keep smooth the road curvatures[2]. The coordinates of the

[2] In classical OCP formulation, where the controls are algebraic variables and their derivatives must not

road right and left borders, $\boldsymbol{b}_r$ and $\boldsymbol{b}_l$, can be expressed as function of the state variables:

$$\begin{aligned}
\boldsymbol{b}_r &= \begin{bmatrix} x_r + \frac{r_w}{2}\left(\cos\psi\sin\sigma\sin\beta - \sin\psi\cos\beta\right) \\ y_r + \frac{r_w}{2}\left(\sin\psi\sin\sigma\sin\beta + \cos\psi\cos\beta\right) \\ z_r + \frac{r_w}{2}\cos\psi\sin\beta \end{bmatrix} \\
\boldsymbol{b}_l &= \begin{bmatrix} x_r - \frac{r_w}{2}\left(\cos\psi\sin\sigma\sin\beta - \sin\psi\cos\beta\right) \\ y_r - \frac{r_w}{2}\left(\sin\psi\sin\sigma\sin\beta + \cos\psi\cos\beta\right) \\ z_r - \frac{r_w}{2}\cos\psi\sin\beta \end{bmatrix}
\end{aligned} \tag{B.7}$$

The solution of the optimal control problem (B.3) allows to find the road middle lane $\mathcal{C}(s)$, orientation $\boldsymbol{O}_r(s)$, and curvatures κ, ν, τ as a function of the curvilinear abscissa s. This formulation allows to remove the noise affecting the input data (3D road map) by weighting the controls in the target function: the greater w_u is, the more the controls are penalised and the more the output data is filtered.

appear, the state space can be augmented by including the controls of problem (B.3) as state variables. Thus the penalty on the control in (B.6b) can then be expressed as a penalty on the control itself.

Appendix C

Motorbike model on a three-dimensional road

The motorbike model used for the U-turn manoeuvre in section 6.4 is here presented. This model does not aim to provide an accurate description of the motorbike dynamics, rather it is intended as a toy model for simulations on a three-dimensional road model. Despite its conciseness, this model captures the essence of a motorbike dynamics (i.e. load transfers, gyroscopic effects and g-g diagram), moreover it highlights the advantages of an implicit optimal control formulation from a modelling perspective.

The model is composed by the chassis main body, that has six degrees of freedom, plus the two wheels. The road model is the $3D$ one described in appendix A. The motorbike is tracked by means of the six variables as described in 6.4: the curvilinear abscissa s, the lateral displacement from the road middle lane n, the heading angle w.r.t the road middle lane α, the roll angle ϕ, the vertical displacement from the nominal CoM height z and the pitch angle μ. Thus the road to vehicle transformation matrix $\boldsymbol{W}_{rel}$ is the one described in section 6.4 (see equation (6.48)).

Newton and Euler equations

The equations of motion for the whole vehicle can be easily expressed as function of the net external forces $\boldsymbol{F}$, torques $\boldsymbol{\tau}$, chassis speed $\boldsymbol{v}_v$ and angular velocity $\boldsymbol{\omega}_v$ through the following equation:

$$\begin{aligned}\boldsymbol{F} =& m_v(\dot{\boldsymbol{v}}_v + \boldsymbol{\omega}_v \times \boldsymbol{v}_v)\\ \boldsymbol{\tau} =& \boldsymbol{I}_v\dot{\boldsymbol{\omega}}_v + \boldsymbol{\omega}_v \times \boldsymbol{I}_v\boldsymbol{\omega}_v + \boldsymbol{I}_w\dot{\boldsymbol{\omega}}_w + \boldsymbol{\omega}_w \times \boldsymbol{I}_w\boldsymbol{\omega}_w\end{aligned} \tag{C.1}$$

where m_v is the total mass (motorcycle plus rider), $\boldsymbol{I}_v$ is the total inertia tensor, $\boldsymbol{I}_w$ is an inertia tensor that includes only the spin inertia of the wheels and ω_w is the spin velocity of the wheels. It can be noted that the Euler equation in (C.1) is the equation of a body (the whole motorcycle) plus a spinning body (wheels) that adds extra gyroscopic effects. Since the wheel spin velocity $\boldsymbol{\omega}_w$ in equation (C.1) contributes only to gyroscopic effects, it is calculated for simplicity in zero-slip conditions, thus it is equal to the ratio between the forward motorcycle speed v_{vx} and the tyre rolling radius r_{tr} (assumed equal for the front and rear wheels):

$$\boldsymbol{\omega}_w = \left[0;\ -\frac{v_{vx}}{r_{tr}};\ 0\right] \tag{C.2}$$

Moreover, the wheel inertia $\boldsymbol{I}_w$ tensor includes both the front $I_w^{(f)}$ and rear $I_w^{(r)}$ wheel spin inertia together with the engine (reduced to wheel) inertia I_e:

$$\boldsymbol{I}_w = \begin{bmatrix} 0 & 0 & 0 \\ 0 & I_w^{(f)} + I_w^{(r)} + I_e & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{C.3}$$

The external forces $\boldsymbol{F}$ and torques $\boldsymbol{\tau}$ are due to the gravity, tyre and aerodynamic forces. Tyre forces include the tyre load N, the longitudinal F_{tx} and lateral F_{ty} forces. Aerodynamic forces are the drag F_d and F_l ones. Thus the external forces and torques can be decomposed as follows:

$$\begin{aligned} \boldsymbol{F} =& \boldsymbol{F}_t^{(r)} + \boldsymbol{F}_t^{(f)} + \boldsymbol{F}_a + \boldsymbol{F}_g \\ \boldsymbol{\tau} =& \boldsymbol{p}_t^{(r)} \times \boldsymbol{F}_t^{(r)} + \boldsymbol{p}_t^{(f)} \times \boldsymbol{F}_t^{(f)} + \boldsymbol{p}_a \times \boldsymbol{F}_a \end{aligned} \tag{C.4}$$

where $\boldsymbol{F}_t^{(r)}$, $\boldsymbol{F}_t^{(f)}$ are the vectors respectively of the rear and front tyre forces, $\boldsymbol{F}_a$ is the vector of the aerodynamic forces and $\boldsymbol{F}_g$ takes into account for the gravity:

$$\boldsymbol{F}_t^{(r)} = \begin{bmatrix} F_{tx}^{(r)} \\ F_{ty}^{(r)} \\ -N^{(r)} \end{bmatrix}, \quad \boldsymbol{F}_t^{(f)} = \begin{bmatrix} F_{ty}^{(f)} \\ F_{ty}^{(f)} \\ -N^{(f)} \end{bmatrix}, \quad \boldsymbol{F}_a = \begin{bmatrix} -F_d \\ 0 \\ -F_l \end{bmatrix}, \quad \boldsymbol{F}_g = mg \begin{bmatrix} F_{gx} \\ F_{gy} \\ F_{gz} \end{bmatrix}$$

$$\begin{aligned} F_{gx} =& \sin(\beta)\cos(\sigma)\cos(\alpha)\sin(\phi)\mu - \cos(\sigma)\cos(\phi)\cos(\beta)\mu \\ &+ \sin(\phi)\sin(\sigma)\sin(\alpha)\mu + \sin(\beta)\cos(\sigma)\sin(\alpha) - \sin(\sigma)\cos(\alpha) \\ F_{gy} =& \sin(\alpha)\cos(\phi)\sin(\sigma) + \cos(\alpha)\cos(\phi)\cos(\sigma)\sin(\beta) \\ &+ \sin(\phi)\cos(\sigma)\cos(\beta) \\ F_{gz} =& \sin(\beta)\cos(\sigma)\sin(\alpha)\mu - \sin(\beta)\cos(\sigma)\cos(\alpha)\sin(\phi) \\ &- \cos(\alpha)\sin(\sigma)\mu - \sin(\sigma)\sin(\alpha)\sin(\phi) + \cos(\phi)\cos(\sigma)\cos(\beta) \end{aligned} \tag{C.5}$$

In the Euler equation (C.4), $p_t^{(r)}$ and $p_t^{(f)}$ are respectively the rear and front tyre contact point coordinates (w.r.t the motorcycle CoM), and $\boldsymbol{p}_a$ are the coordinates of the aerodynamic centre of pressure (w.r.t the motorcycle CoM). The gravity does not generates external net torques on the motorbike.

Aerodynamic forces

Aerodynamic drag F_d and F_l forces are proportional to the square of the speed:

$$F_d = \frac{1}{2}\rho c_{dA} \|v_v\|^2 , \quad F_l = \frac{1}{2}\rho c_{lA} \|v_v\|^2 \tag{C.6}$$

where ρ is the air density, c_{dA} and c_{lA} are respectively the drag and lift coefficients.

Tyre forces

Tyre longitudinal forces ($F_{tx}^{(r)}$, $F_{tx}^{(f)}$) together with the front tyre lateral force $F_{ty}^{(f)}$ are directly controllable by the driver through throttle, braking and steering inputs. Therefore these three quantities $F_{tx}^{(r)}$, $F_{tx}^{(f)}$, $F_{ty}^{(f)}$ are control inputs in the optimal control problem, with the restriction that $F_{tx}^{(r)}$ can be only positive (i.e. braking with the rear wheel is avoided) and $F_{tx}^{(f)}$ can be only negative. Differently, the tyre loads and the rear tyre lateral force are calculated as follows:

$$\begin{aligned} F_{ty}^{(r)} &= N^{(r)} \left(K_{t\lambda}^{(r)} \lambda_t^{(r)} + K_{t\phi}^{(r)} \phi \right) \\ N^{(r)} &= N_0^{(r)} + K_{er}^{(r)} \xi^{(r)} + K_{ed}^{(r)} \dot{\xi}^{(r)} \\ N^{(f)} &= N_0^{(f)} + K_{er}^{(f)} \xi^{(f)} + K_{ed}^{(f)} \dot{\xi}^{(f)} \end{aligned} \tag{C.7}$$

where $K_{t\lambda}^{(r)}$ and $K_{t\phi}^{(r)}$ are respectively the rear tyre sideslip and roll stiffness, N_0 is the tyre load in static conditions and ξ is the tyre radial deformation. Moreover K_{er} and K_{ed} are respectively the equivalent vertical stiffness and damping coefficient and take into account both for the tyre and suspension compliance. In different words, K_{er} is given by the series of the tyre and suspension stiffness (reduced at wheel), and K_{ed} is tuned so as to preserve the damping ratio.

The tyre deformations ξ can be calculated through kinematic considerations, and they are given by:

$$\xi^{(r)} = \cos\phi \, (z + b\mu) \, , \quad \xi^{(f)} = \cos\phi \, (z - a\mu) \tag{C.8}$$

where b and a are respectively the longitudinal distance of the rear and front wheel from the CoM. The expressions in (C.8) have been expanded in Taylor series up to the first order in μ.

The rear tyre sideslip angle $\lambda_t^{(r)}$ depends instead on the rear tyre contact point lateral v_{ty} and longitudinal v_{tx} speed:

$$\begin{aligned}
\lambda_t^{(r)} =& -\arctan\left(\frac{v_{ty}}{v_{tx}}\right) \\
v_{tx} =& v_{vx} - r_t\dot{\mu} + (\mu b + h)\,\omega_{vy} + \mu v_{vz} \\
v_{ty} =& \sin\phi\,\big(\mu\omega_{vy}(h - r_t) + v_{vx}\big) - \omega_{vy} b - v_{vz}\big) \\
& - \cos\phi\,\big(\mu\omega_{vz}(t) r_t + b\omega_{vz} + h\omega_{vx} - v_{vy}\big)
\end{aligned} \tag{C.9}$$

The rear tyre contact point lateral v_{ty} and longitudinal v_{tx} speeds have been expanded in Taylor series up to the first order in μ.

Finally, tyre forces are limited by an adherence ellipse as follows:

$$\left(\frac{F_{tx}^{(r)}}{N^{(r)}\mu_x^{(r)}}\right)^2 + \left(\frac{F_{ty}^{(r)}}{N^{(r)}\mu_y^{(r)}}\right)^2 \leq 1, \qquad \left(\frac{F_{tx}^{(f)}}{N^{(f)}\mu_x^{(r)}}\right)^2 + \left(\frac{F_{ty}^{(f)}}{N^{(f)}\mu_y^{(r)}}\right)^2 \leq 1 \tag{C.10}$$

where μ_x and μ_y are respectively the tyre longitudinal and lateral maximum adherence.

State space

The motorcycle model here presented has six degrees of freedom, thus 12 variables are required to describe it by a set of first order equations. Six of the 12 variables are the tracking variables that determine the position and orientation of the motorcycle in the space. These six variables are: s, n, z, α, ϕ, and μ. The other six velocity variables are instead the speed and angular velocity of the motorcycle: v_{vx}, v_{vy}, v_{vz}, ω_{vx}, ω_{vy}, ω_{vz}. The six first order equations related to the position variables are equations (6.50), while the other six equations are the Newton and Euler ones (C.4) (once the expressions of the external forces have been substituted into them). Controls rely on the tyre longitudinal forces and on the front tyre lateral force: $F_{tx}^{(r)}$, $F_{tx}^{(f)}$, $F_{ty}^{(f)}$.

It should be noted that equations (C.4) are not linear in the state space variable derivatives. Indeed, both the rear tyre load $N^{(r)}$ and the rear tyre sideslip angle $\lambda_t^{(r)}$ depend on state variable derivatives (the former through (C.8), the latter through (C.9)). The rear tyre lateral force $F_{ty}^{(r)}$ as expressed in (C.7) depends on the product of $N^{(r)}$ by $\lambda_t^{(r)}$, thus it is quadratic in the state variable derivatives, at least with the formulation here adopted. For this reason the first order equations of this simple motorcycle model cannot be written neither in the explicit form (6.1), nor in the linear form (6.2). An implicit OCP formulation allows instead to straightforwardly use this model for optimal control simulations.