POLITECNICO DI TORINO

Master's Degree in Automotive Engineering

Master's Degree Thesis

An Economic Model Predictive Control Approach for Adaptive Cruise Control Design in Full Electric Vehicles

Supervisors

Prof. Angelo BONFITTO

Prof. Michele PAGONE

Candidate

Mattia ACAMPORA

March 2025

Summary

In the ever-evolving scenario of present days, the road transport industry is in the middle of a period of great changes. The urgent need for more energy-efficient transport has led to significant advancements in the introduction of vehicles powered by cleaner energy forms than fossil fuels: the Electric Vehicles (EVs).

Simultaneously, road safety remains a critical concern, where the human factor represents the primary cause of traffic accidents. So, the increasing demand for safer solutions has driven the automotive industry toward the development of autonomous driving vehicles and driving assistance systems.

Autonomous driving, combined with electric propulsion, represents a promising future solution to these challenges by reducing road accidents and optimizing energy consumption. However, the introduction of electric autonomous vehicles leads to new technical challenges, particularly in terms of efficient energy management and precise vehicle control.

This thesis specifically addresses these challenges by focusing on the development of a Nonlinear Model Predictive Control (NMPC) strategy with the aim of improving both driving safety and energy efficiency.

Traditional controllers like PID and LQR have inherent limitations: the PID is not an optimal control strategy, while the LQR, despite being optimal, is limited by its linear nature, making it unsuitable for handling nonlinear dynamics and constraints on states and inputs. In contrast, NMPC predicts and optimizes future control actions while explicitly considering physical and safety constraints, making it particularly well-suited for autonomous driving applications.

In order to obtain an NMPC controller able to predict the behavior of the vehicle and suggest the best control sequence, such that a safe and energy-efficient scenario is achieved, a detailed vehicle model is developed by incorporating the vehicle longitudinal dynamics and the electric powertrain characteristics. Other than system model, the NMPC controllers also need to considers accurate real-world constraints, such as battery SOC limits, motor power capabilities, and maximum allowable speed, that allow the NMPC to ensure feasibility and efficiency in real-time applications.

In this thesis, NMPC has been chosen as suitable control tool for designing an

optimal Adaptive Cruise Control. The NMPC- based ACC performances have been compared with respect to traditionally used controller, the Constant-Time-Gap (CTG) controller, producing good results in terms of both driving safety and energy efficiency.

The aim of this work is to contribute to the development of control strategies for future electric autonomous vehicles, in particular by focusing on the energy saving, that may be seen as one of the main concerns that limits the shift to electric vehicles, while not losing sight of safety, that remains a fundamental objective.

Table of Contents

List of Tables

List of Figures

Acronyms

EV
Electric Vehicles

HEV
Hybrid Electric Vehicles

ICE
Internal Combustion Engine

PID
Proportional Integral Derivative

LQR
Linear Quadratic Regulator

MPC
Model Predictive Control

NMPC
Non-linear Model Predictive Control

ACC
Adaptive Cruise Control

SOC
State of Charge

CTG
Constant Time Gap

Chapter 1

Introduction

In the last decades, the automotive industry has undergone significant transformations. Two major innovations have reshaped the sector: the rise of electric and hybrid powertrains and the development of autonomous driving vehicles. These advancements have been driven by both environmental concerns and the pursuit of safer and more efficient transportation solutions.

The first major shift, the transition toward electrification, arises from the need to address environmental issues, particularly air pollution. As shown in Figure 1.1, the contribution of road transport to global CO_2 emissions is significant, and this is mainly due to the widespread use of internal combustion engines (ICE). To address this issue, the European Union has set ambitious targets, aiming for a 100% reduction in CO_2 emissions from passenger and light-duty vehicles by 2035 [1]. In response, automotive manufacturers have accelerated the production of hybrid (combining internal combustion and electric powertrains) and fully electric vehicles (EVs), where electric motors convert the energy stored in the battery into mechanical power.

Despite their environmental benefits, electric vehicles face a major challenge: limited range, i.e. the maximum distance a vehicle can travel on a single battery charge. This limitation has been a key factor limiting their widespread adoption, forcing manufacturers and researchers to focus on optimizing energy consumption to maximize vehicle range.

The second crucial transformation in the automotive industry is the development of autonomous driving technologies. Modern vehicles already feature varying levels of automation, from adaptive cruise control to advanced driver assistance systems (ADAS). However, fully autonomous vehicles are not yet widely available, particularly in Europe, whereas in some regions of North America, they are already in commercial use.

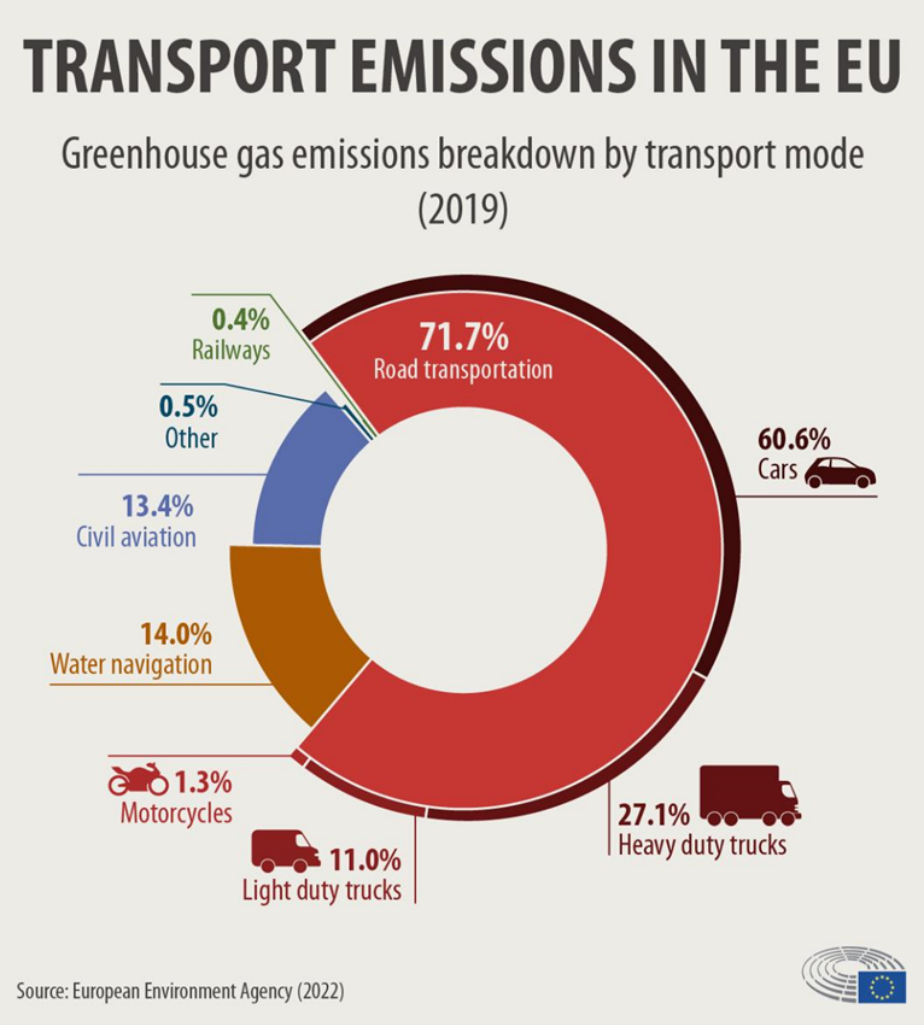

Figure 1.1: Road transport emission with respect to total transport emissions

Source: European Environment Agency,
url: `https://www.europarl.europa.eu/topics/en/article/20190313STO31218/co2-emissions-from-cars-facts-and-figures-infographics`

The push toward autonomous driving is largely motivated by road safety concerns. A substantial percentage of traffic accidents are caused by human error. By delegating vehicle control, on braking, steering, acceleration, and gear selection, to advanced algorithms, autonomous vehicles have the potential to significantly reduce accident rates. Although these technologies are still being refined, early data from the deployment of autonomous vehicles in the U.S. suggests promising safety improvements.

Given this evolving landscape, this thesis focuses on the development and implementation of a Nonlinear Model Predictive Control (NMPC) strategy for the autonomous driving of an electric vehicle. Particular emphasis will be placed on safety and energy efficiency, with the ultimate goal of maximizing vehicle operational range while ensuring robust and reliable autonomous navigation.

1.1 Objectives

The main objective of this Master Thesis is to elucidate the functionality of a **Nonlinear Model Predictive Control**, applied in the automotive field. This aim will be pursued by using the control strategy to predict the states of the longitudinal dynamics of a vehicle with an electric powertrain, with the objective of minimizing the battery consumption when following a given speed profile (WLTP3).

Furthermore, in order to highlight the advantages of the Nonlinear Model Predictive Controller with respect to other kind of controller, the studied controller results will be compared with the results given by a Constant-Time-Gap (CTG) Controller on the same mission.

To summarize, this research aims to:

- Analyze the dynamic model of the vehicle and of its electric powertrain, highlighting the main non-linear aspects that influence the control.
- Formulate the NMPC control problem, considering physical constraints, input limits and an appropriate objective function.
- Implement and simulate the controller in a software environment, in particular in MATLAB environment by exploiting CasADi functionalities.
- Evaluate the performance of the NMPC through comparative tests with other controllers, such as a Constant-Time-Gap controller.
- Identify potential improvements and future developments, especially in terms of computational optimization and implementation on real hardware.

1.2 Thesis structure

The present work will be organized in the following chapters:

- **Chapter 2 - State of the Art:** An overview of the main control techniques in autonomous vehicles is presented, with particular focus on the Model Predictive Control.

- **Chapter 3 – Vehicle Model:** The dynamics of the vehicle and the operations of its electric powertrain are analyzed and the system equations needed in the controller design are defined.
- **Chapter 4 – Nonlinear Model Predictive Control (NMPC):** The methodology adopted for designing the controller is described, by specifying the numerical equations, the constraints and the cost function that define the optimization problem.
- **Chapter 5 - Implementation and Simulation:** The exploited simulation tools and the tests performed to assess NMPC performances are described in details.
- **Chapter 6 - Simulation results:** The results obtained from performed simulations are accurately analyzed in order to gain a deep understanding of the performances of the NMPC.
- **Chapter 7 – Conclusions:** The main results of the whole analyses are summarized and possible future extensions of the work are discussed.

Chapter 2

State of Art

Before entering into the details of vehicle and controller model development, an intensive research had been carried out about the state of the art in autonomous driving and especially about the types of Model Predictive Controllers used today in the automotive industry and in research field.

This chapter aims to provide an overview of the main existing technologies related to autonomous vehicle control and, in particular, Nonlinear Model Predictive Control (NMPC).

2.1 Literature Review

The automotive industry has undergone significant transformation in recent years, mainly driven by the need for energy-efficient and safe motion and also by the surprising advances in autonomous driving technology.

Many have been the autonomous control strategies that researchers and manufacturers have analyzed in order to develop a smart controller that would be capable of accurately driving a vehicle by responding promptly to any sudden change in the surrounding 'environment'.

The first two control technologies to be widely used for vehicle control applications are the Proportional-Integral-Derivative (PID) controllers and Linear Quadratic Regulators (LQR) controllers.

The PID controller operate based on a feedback loop, where the control input is determined by considering the proportional, integral and derivative terms of the error signal. They have historically been largely used in automotive applications due to their simplicity and effectiveness. For instance, PID controller have been successfully implemented in Adaptive Cruise Control systems to enhance vehicle safety by maintaining a desired speed or by safely following a certain distance from

the preceding vehicle [2]. Additionally, Jain et al. (2023) demonstrated that PID controller can be well applied not only for vehicle longitudinal dynamics, but also for lateral dynamics control [3]. However, the main drawback of PID controllers is their lack of adaptability to system changing conditions, in fact they need to be accurately tuned for different operating scenarios, and they struggle with handling non-linearities, that are typical of autonomous driving applications.

To address some of the PID controllers limitations, Linear Quadratic Regulators (LQR) controllers have been introduced as a control approach for linear systems. In particular, LQR minimizes a quadratic cost function that balances tracking performance and control effort. This kind of control strategy makes it well suited for managing the yaw rate of the vehicle to ensure the stability during different driving conditions [4]. Furthermore, LQR controllers have been also exploited in active suspension systems to optimize vehicle body displacement, in order to improve riding comfort and handling [5]. However, LQR assumes to deal with a linear system model, which limits its applicability to real-world driving conditions where vehicle dynamics are strongly non-linear.

Both PID and LQR controllers have proven to be efficient in some specific scenarios, but they have also shown some important limitations in handling the non-linear and highly dynamic nature of autonomous driving operations. For those reasons, researches and manufacturers have explored different control strategies such as Model Predictive Control (MPC).

MPC is an optimization-based control strategy that predicts the future behavior of the system over a finite time horizon and solves an optimal control problem at each step. This allows to explicitly handle the system constraints and the anticipation of future events, making it well-suited for dynamic and uncertain scenarios encountered in autonomous driving. Traditionally, Linear MPC have been successfully implemented in autonomous vehicle control for many different applications: for example, longitudinal and lateral applications [6] and steering control [7], but also path tracking and obstacle avoidance [8]. Unfortunately, due to the typical non-linearities of vehicle dynamics, especially under high-speed maneuvers or varying road conditions, the Linear MPC controllers have shown some limitations in accurately predicting vehicle behavior.

To solve the issue related to the limitations of Linear MPC, researchers have developed Nonlinear Predictive Control (NMPC), which integrates non-linear vehicle models to improve the accuracy and performances of the controller. Many studies have demonstrated the effectiveness of the NMPC implementation in autonomous driving applications. Falcone et al. (2007) introduced an NMPC designed to account for the nonlinearities of vehicle dynamics, improving trajectory tracking

capabilities compared to traditional methods [9]. Di Cairano et al. (2013) further extended NMPC to vehicle stability control, where the model effectively controls nonlinear tire-road interactions, ensuring stability in high-speed driving scenarios [10]. Additionally, Rosolia and Borrelli (2017) explored a peculiar NMPC approach, where the iterative learning was used to refine the control strategies over multiple driving cycles, leading to improved performance in real-world driving conditions[11]. Brown et al. (2020), instead, investigated the application of NMPC for obstacle avoidance, demonstrating its capability to dynamically adapt to dynamic scenarios while maintaining its stability [12].

Despite its advantages, the practical implementation of NMPC leads to several problems. One of the most significant obstacles is the computational complexity: in fact, solving non-linear optimization problems in real-time requires an extremely high computational power. Another critical challenge is the model accuracy, in fact, NMPC relies on the accuracy of the vehicle model that predicts future states, so too simplified models may lead to non-optimal performances, while too complex models may increase the computational demand. Moreover, the NMPC response to uncertainties remains a key concern, particularly in real-world applications where sensor noise, road conditions and/or external disturbances may affect system performances.

Nowadays, the NMPC continues to be an important subject in autonomous vehicle research, offering a flexible and effective solution for real-time vehicle control, despite computational limitations and robustness concerns remain research fields. The current trend in terms of new researches aims to integrate NMPC algorithms with emerging AI-driven techniques to enhance performance.

As autonomous vehicles are currently getting closer and closer to be approved for market release, the continued development of NMPC-based control systems will play a crucial role in ensuring safe, efficient, and reliable autonomous driving.

Chapter 3

Vehicle Model

As it has been stated previously, the performances of the NMPC are strictly related also to the system model, exploited to predict future states, that is considered: a too simplified model may lead to not really accurate results, while a too complex model may produce an increase in the computational cost for the controller. So, the modeling of the system in simulation environment becomes crucial for the operations of the NMPC.

In the present work, the NMPC has been developed to control the dynamic of an electrified vehicle, so in this chapter an overview of the longitudinal dynamic laws of the vehicle, of the main components of the electric vehicle powertrain and of the way in which they have been modeled will be provided.

3.1 Longitudinal Dynamics Model

The longitudinal dynamics model represents the forces acting on the vehicle on the x-axis direction (the axis on which it is moving). In particular, this dynamic describes the relationship between the applied traction force, given by the motor through the wheels) and the resulting vehicle motion, observed at the vehicle center of gravity. The difference between those two quantities is given by the resistance that different components put up to the vehicle motion.

The vehicle motion, or acceleration, is computed based on the Newton's Second Law as:

$$m\ \dot{v_x} = F_{\text{trac}} - (F_{\text{roll}} + F_{\text{aero}} + F_{\text{slope}}) \tag{3.1}$$

where:

- m is the vehicle mass;
- v_x is the longitudinal velocity, so $\dot{v_x}$ is the longitudinal acceleration;

- F_{trac} is the traction force provided by the powertrain;
- F_{roll} is the rolling resistance force;
- F_{aero} is the aerodynamic resistance force;
- F_{slope} is the gravitational force due to road inclination (slope).

The next paragraphs provide an explanation of the resistance forces that influence the longitudinal dynamics of the vehicle: the Rolling Resistance, the Aerodynamic Resistance, and the Slope Resistance.

These forces play a crucial role in the energy consumption of the vehicle and must be taken into account when implementing NMPC control to ensure optimal efficiency.

3.1.1 Rolling resistance

The rolling resistance force (F_{roll}) arises due to the deformation of tires as they roll on the road surface. This deformation leads to some change in the pressure distribution, and consequently to some energy dissipation, that results in a resistive force that opposes to vehicle motion. Rolling resistance is one of the major sources of energy loss in vehicles, especially at low speeds.

The rolling resistance force is typically modeled as :

$$F_{\text{roll}} = m\ g\ f_r \cos(\theta) \tag{3.2}$$

where:

- m is the vehicle mass;
- g is the gravitational acceleration;
- f_r is the rolling resistance coefficient;
- θ is the road inclination angle.

In particular, the rolling resistance coefficient f_r depends on several factors: tire type and material, tire pressure and road surface.

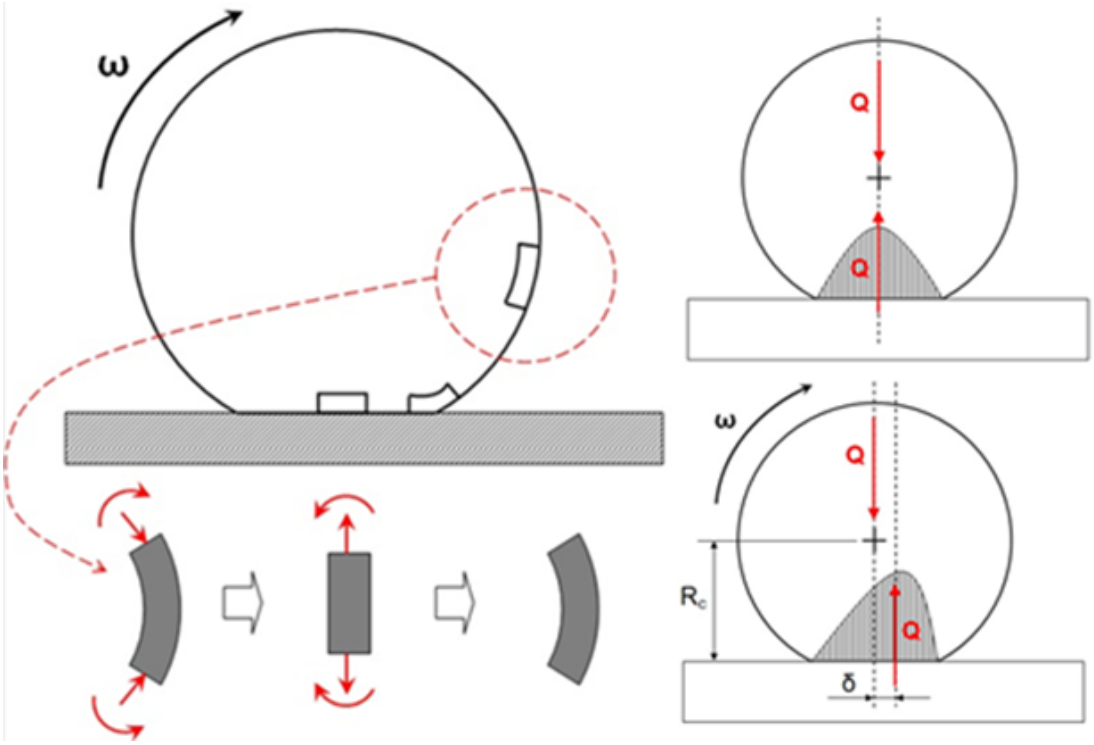

Figure 3.1: Rolling resistance principle schematization

Source: meccanicadelveicolo.com,
url: `meccanicadelveicolo.com/2019/01/07/la-resistenza-al-rotolamento/`

3.1.2 Aerodynamic resistance

The aerodynamic resistance, also called Drag Force (F_{aero}) is a force that arises in opposition to vehicle motion and is caused by the air pushing against the vehicle as it moves forward. This force largely increases with speed, so it is the dominant component of the vehicle motion resistance when vehicle travels at high velocities.

The aerodynamic resistance force is modeled by the following equation:

$$F_{\text{aero}} = \frac{1}{2}\rho A_f C_d v_x^2 \tag{3.3}$$

where:

- ρ is is the air density (typically $1.225 kg/m^3$ at sea level);
- A_f is the frontal area of the vehicle;
- C_d is the drag coefficient;

- v_x is the longitudinal velocity of the vehicle.

Most of the focus is usually given to the drag coefficient since it is a dimensionless parameter that quantifies the aerodynamic efficiency of the vehicle.

Figure 3.2: Aerodynamic resistance principle schematization

Source: scienceabc.com,
url: `https://www.scienceabc.com/eyeopeners/why-do-cars-undergo-wind-tunnel-testing.html`

3.1.3 Slope resistance

The slope resistance, also known as gravitational resistance, represents the component of vehicle's weight that acts along the direction of motion when traveling on an inclined road. This force plays a significant role especially in determining the required traction force needed to travel on a extremely steep road.

By considering the schematization in figure below, when a vehicle travels on a road with an inclination angle θ (considered positive for uphill roads and negative for downhill roads), the gravitational force can be decomposed in two components:

- A vertical component, that is perpendicular to the road and to vehicle motion and contributes to the normal force on wheels;
- A longitudinal component, that acts on a direction that is parallel to the road, and that opposes to vehicle motion in case of uphill roads or assists it in downhill scenarios.

The slope resistance force, that corresponds to the longitudinal component of the gravitational force, is given by:

$$F_{\text{slope}} = m\ g\ \sin\theta \tag{3.4}$$

where:

- m is the vehicle mass;
- g is the gravitational acceleration;
- θ is the road inclination angle.

The value of the road angle is the major component in this equations since according to its value the effect of the slope force significantly changes:

- If $\theta > 0$ (uphill), the F_{slope} acts against vehicle's motion, increasing the power needed from the powertrain to move the vehicle;
- If $\theta < 0$ (downhill), the F_{slope} acts in the direction of vehicle motion, meaning that the vehicle may increase its speed without additional propulsion from the powertrain, possibly requiring some braking force to maintain a safe control of vehicle speed.

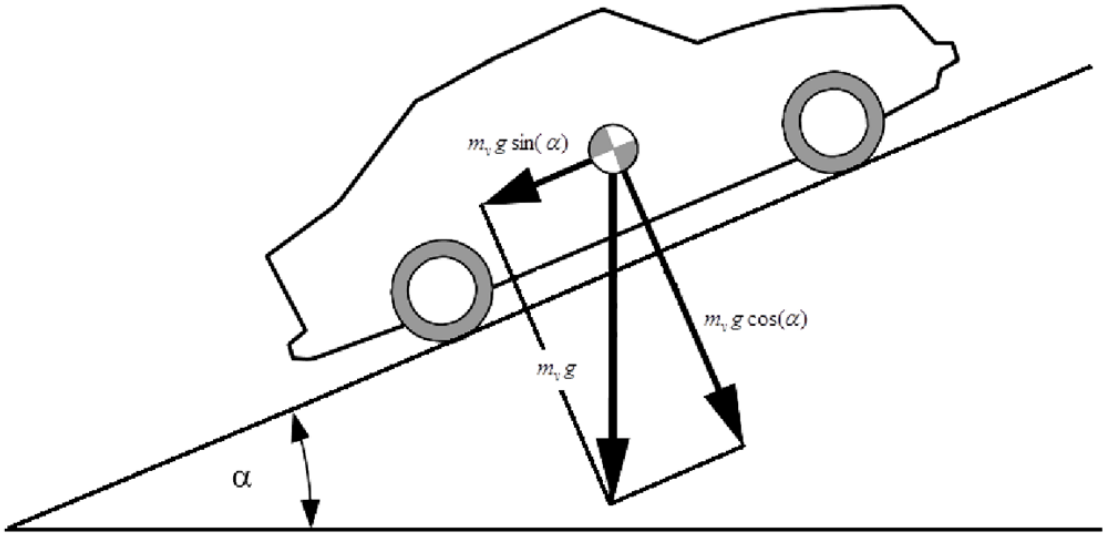

Figure 3.3: Slope resistance principle schematization

Source: Hermansson, V., and Moparthi, K. (2016). Control of an electric vehicle powertrain to mitigate shunt and shuffle [13]

3.2 Electric Vehicle Powertrain Model

In this section, the aim is to provide an overview of the electric powertrain architecture and what is the "dynamic" that regulates its operations.

The powertrain of an electric vehicle [14] is composed by:

- The **Power Source**, that is a crucial component since it provides the necessary energy to propel the vehicle. The most used power sources are battery cells, usually lithium-ion batteries, due to their high energy density and long cycle life. Other power sources, like fuel cells, are also being explored;

- The **Battery Management System (BMS)**, that is responsible for monitoring and controlling the operations of the battery pack ensuring good levels of performance, efficiency and safety;

- The **Traction Inverter** is the electronic component responsible of converting the DC output from the battery into a AC input for the electric machine;

- The **DC-DC converter**, instead, does not change the "nature" of the signal itself, but it steps down the DC voltage from the battery to a much smaller value to be used by auxiliary electronics in vehicles (such as air conditioning, infotainment, etc);

- The **Electric Motor** may be considered as the heart of the electric powertrain, it is responsible for converting the electrical energy coming from the battery into the mechanical energy needed at the wheels to move the vehicle. Various types of electric motors can be used in an electric powertrain, including permanent magnet synchronous motors, induction motors, and switched reluctance motors.
  The electric machine also works as a generator when the torque is negative (breaking maneuver) making it possible to generate power and charge the battery - regenerative breaking;

- The **Transmission system** is responsible for transferring the mechanical power from the electric motor to the wheels, may be single-speed or multi-speed and plays a crucial role in optimizing the power and torque delivery to the wheels, ensuring efficient and smooth acceleration of the vehicle;

- The **On-Board Charger** is responsible for converting the AC power from an external power source, such as a charging station, into DC power to charge the vehicle's battery.

These components work together to provide the necessary power and control for the operation of the electric vehicle.

The figure below shows a generic architecture of a electric vehicle powertrain.

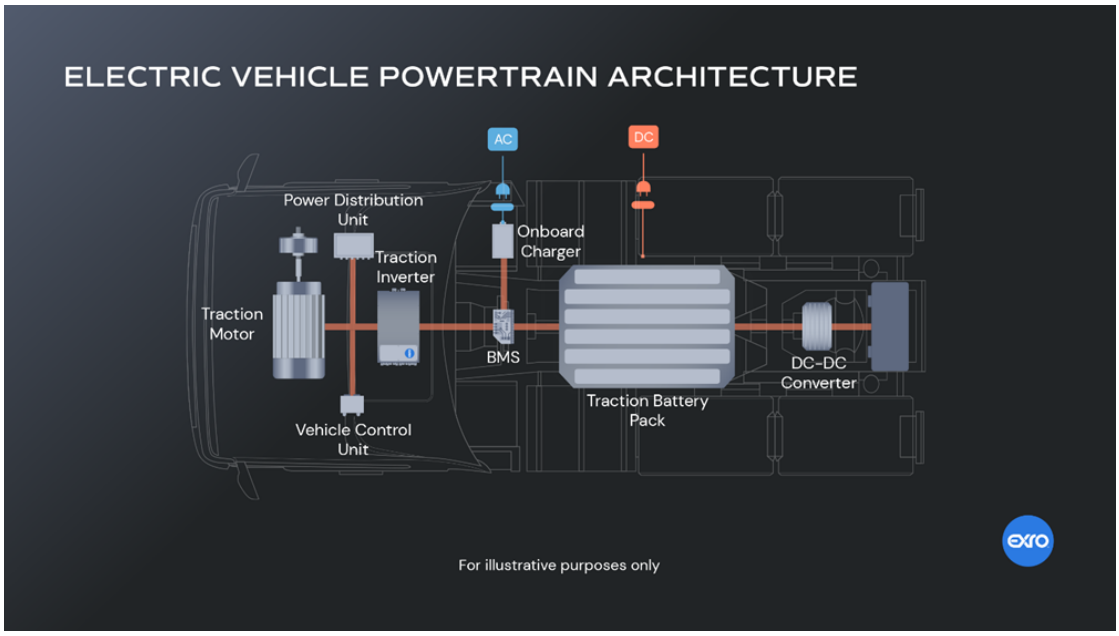

Figure 3.4: Electric Vehicle Powertrain architecture

Source: Exro.com website,
url: `www.exro.com/industry-insights/ev-power-electronics-explained`

3.2.1 Battery Model

The mathematical model of the battery has a crucial role in the understanding of the relation between the power that is provided by the battery to the wheel and the State of Charge (SOC) of the battery itself. This importance is due to the fact that the State of Charge of the battery is essentially the correspondent of the fuel level in a traditional combustion engine vehicle: it gives an idea of the autonomy, so distance that can be covered without the need for refueling or recharging the battery.

As previously stated, the autonomy of the vehicle is the major issue to be assessed when designing an electric vehicle, so the understanding of the manners in which the battery provides energy to the powertrain and then to the wheels is fundamental in order to improve the vehicle's range.

The battery model accurately relates the power that is delivered by the battery to the State of Charge (SOC) of the battery at the present time instant. In particular, the SOC is defined as the ratio between the battery charge Q_b, the amount of energy available inside the battery, and the nominal battery capacity Q_{nom}, the maximum amount of energy that can be stored inside the battery of the vehicle:

$$SOC = \frac{Q_b}{Q_{\text{nom}}} \in [0,1] \tag{3.5}$$

Differentiating both sides it is possible to obtain the derivative of the SOC as a function of the current flowing through the battery I_b:

$$\dot{SOC} = -\frac{1}{\eta_c^{\text{sign}(I_b)}} \frac{I_b}{Q_{\text{nom}}} \tag{3.6}$$

Where Ib is the battery current, Qnom is the nominal capacity and η_c is the coulombic efficiency, that quantifies the fraction of current that is lost during battery charge and discharge [15].

In particular, when the battery current has a positive, it is in discharge mode, so the energy is flowing from the battery to the wheels; while when it has a negative value, it is in charge mode, so the energy that is generated at the wheel when performing a regenerative-braking is fed to the battery.

The battery model in consideration is composed of Nb battery cells connected in series. Each cell is modelled as an ideal voltage source Voc,s with a series output resistance Rob, so considering the whole battery system $V_{oc} = N_b\ V_{oc,s}$ and $R_{o,b} = N_b\ R_{o,s}$ respectively. The values of V_{oc} and $R_{o,b}$ are generally functions of the battery State of Charge. In the present model, such dependencies are derived from the experimental data considered from [16] and interpolated. The results of the interpolations are reported in the figure below.

By considering this model of the battery, the battery power can be related to the battery current, and so to the battery SOC, by the equation:

$$I_b = \frac{V_{\text{oc,b}} - \sqrt{V_{\text{oc,b}}^2 - 4R_{\text{o,b}}P_{\text{batt}}}}{2R_{\text{o,b}}} \tag{3.7}$$

Solving the two equation (3.6) and (3.7), finally a relation between the power provided by the battery to the powertrain and the battery State of Charge is obtained.

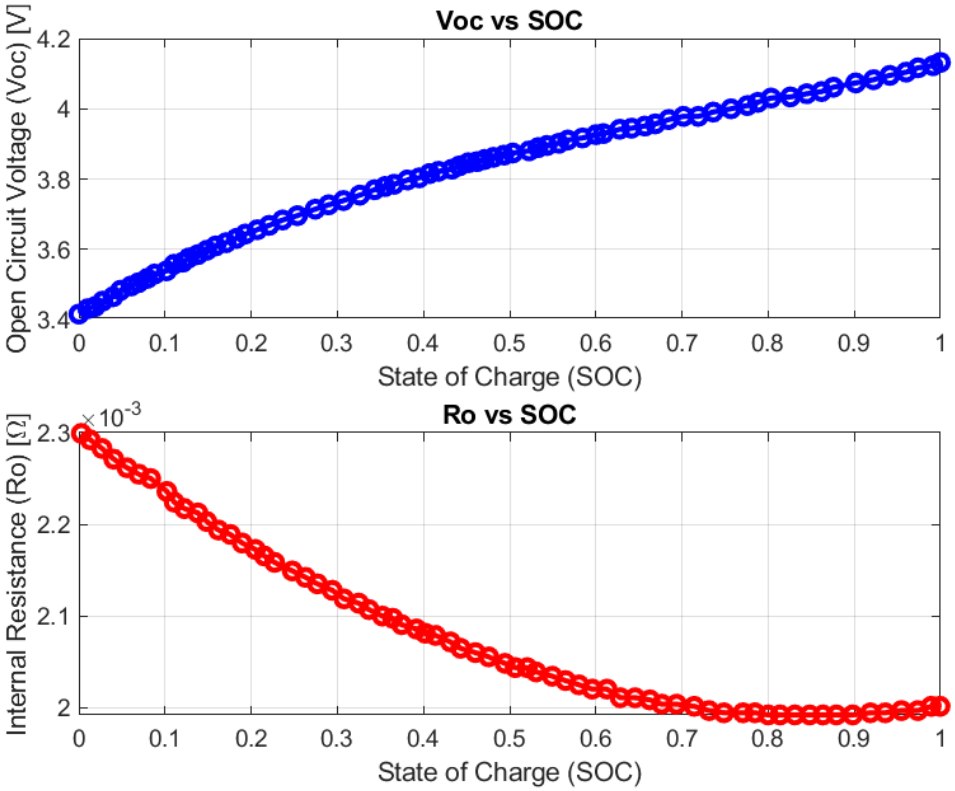

Figure 3.5: Interpolated values of Voc and Ro in function of SOC values

3.2.2 Electric Motor Model

As previously explained, the main purpose of the electric motor is to convert the electric power that is provided in input by the battery to a mechanical power in output to be fed to wheels.

In particular, the power that the electric motor provides is given by a torque component T_{EM} and an angular speed component w_{EM} of the engine shaft, so the value of the power provided or needed by the electric motor is given by the equation:

$$P_{\mathrm{EM}} = T_{\mathrm{EM}} \cdot \omega_{\mathrm{EM}} \tag{3.8}$$

The relation between this value of power P_{EM} and the power provided by the battery P_b essentially depends on the value of the efficiency of the electric motor η_{EM}. This efficiency intrinsically depends on the values of angular speed and shaft

torque at which the electric motor is operating. So, the relationship between P_{EM} and P_b can be obtained by:

$$P_b = \frac{P_{\text{EM}}}{[\eta_{\text{EM}}(\omega_{\text{EM}}, T_{\text{EM}}) \cdot \eta_{\text{inv}}]^{\text{sign}(P_{\text{EM}})}} \tag{3.9}$$

In order to obtain an electric motor model as complete as possible, the value of the efficiency has been mapped with respect to experimental values. The electric motor modeling, therefore, is mainly dependent on the efficiency mapping based on a given angular speed and shaft torque. The resulting efficiency map for the electric motor is represented in the following figures, in 2D and in 3D respectively.

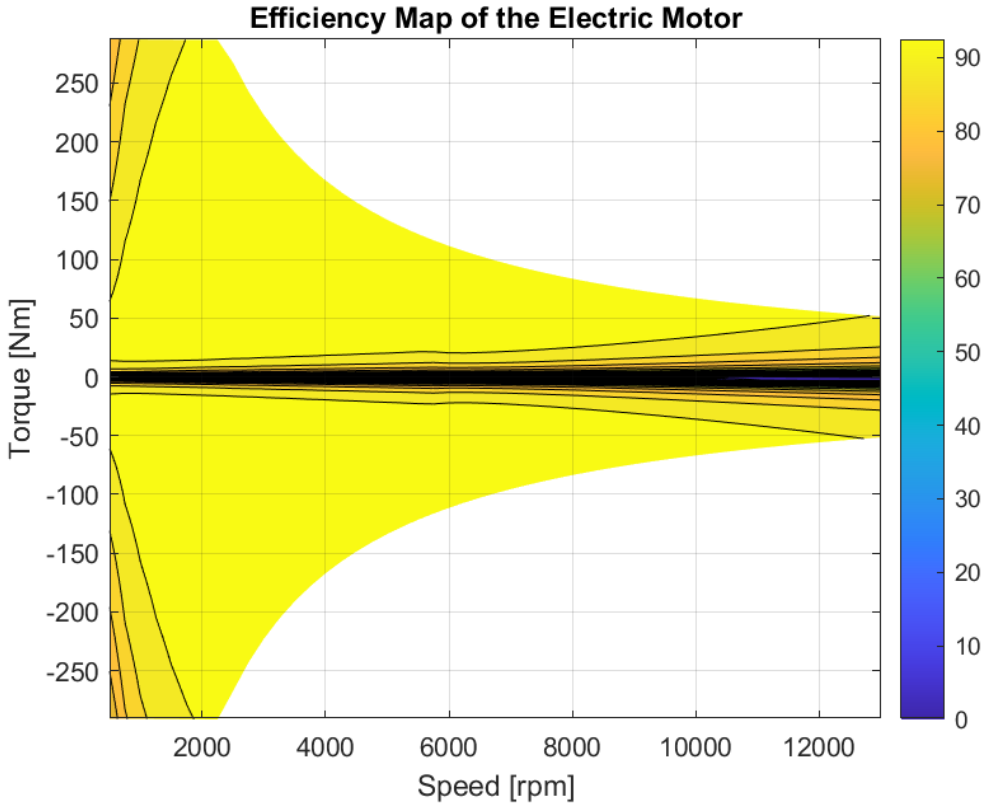

Figure 3.6: 2D representation of the Efficiency Map

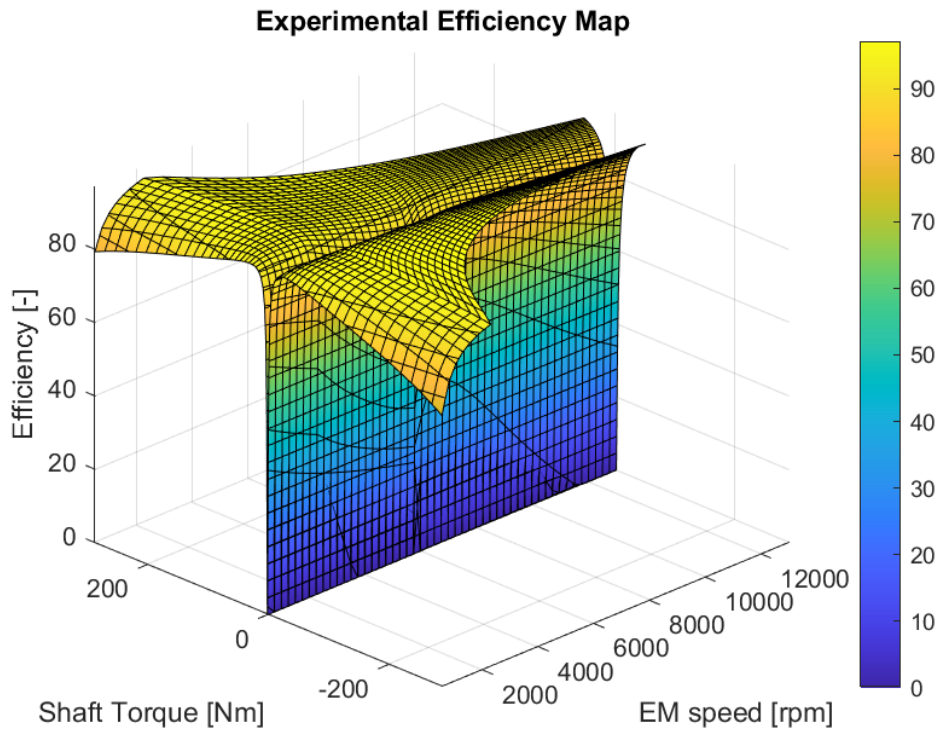

Figure 3.7: 3D representation of the Efficiency Map

3.2.3 Gearbox Model

In the context of an electric powertrain the role of the gearbox element is different with respect to the case of internal combustion engines (ICE). For this reason, while in ICEs multi-speed transmissions are operated, in electric vehicles a single-speed reduction gearbox is employed. In electric vehicles the gearbox has dual purpose:

- Speed matching: The gearbox adjusts the relationship between the motor's rotational speed (w_{EM}) and the wheel speed (w_{wheel}), ensuring efficient power transfer;
- Torque amplification: The gearbox reduces the high-speed, low-torque output of the electric motor and converts it into a higher torque at the wheels, ensuring smooth acceleration.

Given this, it can be stated that the well-designed gearbox can minimize energy losses and improve overall efficiency of the powertrain.

For an electric vehicle single-speed transmission, the gearbox can be modeled as

a fixed-ratio reduction system, characterized by a gear ratio τ_{gb}. Obviously, also the gearbox element introduces some losses in terms of power transmitted, so also the gearbox efficiency η_{tr} has to be introduced.

Now the relationships between the electric motor angular velocity w_{EM} and torque T_{EM} and the wheel angular velocity w_{wheel} and torque T_{wheel} are given by:

$$\omega_{\mathrm{EM}} = \omega_{\mathrm{wheel}} \cdot \tau_{\mathrm{gb}} \tag{3.10}$$

$$T_{\mathrm{EM}} = \frac{T_{\mathrm{wheel}}}{\eta_{\mathrm{tr}}^{\mathrm{sign}(T_{\mathrm{wheel}})} \cdot \tau_{\mathrm{gb}}} \tag{3.11}$$

3.2.4 Wheel Model

When modeling a vehicle, it is fundamental to model also the wheel dynamics, since wheels play a crucial role in translating the torque from the powertrain into linear motion of the vehicle. In fact, the force that is transmitted by the powertrain to the wheels and through the wheels significantly affects vehicle acceleration, traction and overall driving dynamics.

In the wheel model taken into consideration, the force at the wheel-ground contact F_{wheel} and the vehicle speed v_{veh} are respectively related to the torque at the wheel T_{wheel} and to the wheel angular speed w_{wheel} by the wheel radius r_w:

$$F_{\mathrm{wheel}} = \frac{T_{\mathrm{wheel}}}{r_w} \tag{3.12}$$

$$v_{\mathrm{veh}} = w_{\mathrm{wheel}} \cdot r_w \tag{3.13}$$

3.3 Vehicle and Electric Powertrain model parameters

To accurately simulate the behavior of the vehicle and its electric powertrain, several parameters have been defined based on realistic assumptions and existing literature. These parameters play a crucial role in modeling the longitudinal vehicle dynamics, electric motor performance, and battery characteristics.

The vehicle parameters have been selected considering a standard passenger electric vehicle, in particular a FIAT 500e, and are reported in the Table 3.1 below:

Parameter	Symbol	Value	Unit
Vehicle Mass	M_{veh}	1400	kg
Front axle - CoG distance	a	1	m
Rear axle - CoG distance	b	1.3	m
CoG Height	h	0.3	m
Wheel radius	r_w	0.3	m
Frontal Area	A_f	2.15	m^2
Drag coefficient	C_d	0.33	-
Static rolling resistance coefficient	f_0	4.5	N/kN
Dynamic rolling resistance coefficient	k	0.013	Ns/m

Table 3.1: Vehicle parameters

Source: Raffaele Manca et al. [17]

In the Table 3.2, instead, the Electric Powertraind and Battery parameters values are reported:

Parameter	Symbol	Value	Unit
Number of series cells	N_s	108	-
Number of parallel cells	N_p	1	-
Number of total cells	N_b	108	-
Battery nominal capacity	Q_{nom}	60	Ah
Coloumbic efficiency	η_c	0.95	-
Transmission system efficiency	η_{tr}	0.95	-
Inverter efficiency	η_{inv}	1	-
Gear ratio	τ_{gb}	9.6	-
Gearbox efficiency	η_{gb}	0.97	-

Table 3.2: Electric Powertrain and Battery parameters

Source: Raffaele Manca et al. [17]

Chapter 4

Nonlinear Model Predictive Control (NMPC)

4.1 Introduction to NMPC

The high complexity of autonomous driving systems demands advanced control strategies that can handle the highly non-linear vehicle dynamics and the required high-speed decision-making. Among various control methodologies, Model Predictive Control (MPC) has emerged as a powerful solution due to its ability to predict future states, optimize control actions, and respect system constraints.

Model Predictive Control is, in particular, a control strategy widely exploited in various application fields, including the control of electric vehicle powertrains. This control strategy takes into account the dynamic nature of the system and predicts future states and control inputs based on a mathematical model of the system. By explicitly considering the system dynamics, MPC is able to optimize control inputs over a finite time horizon, called prediction horizon, to achieve desired objectives such as energy efficiency, performance, and constraint satisfaction.

Model Predictive Control is capable of dealing with complex multi-input multi-output systems with hard state and input constraints, making it suitable for controlling electric vehicle powertrains with longitudinal dynamics [18].

Furthermore, the ability of MPC to continuously update the model and control strategy allows it to handle changes in system parameters, such as variations in battery capacity or motor efficiency, or even in external environment, like road or weather conditions, without the need for extensive re-calibration.

In particular, while Linear MPC (LMPC) is commonly used in control applications, its effectiveness is limited when dealing with highly nonlinear phenomenon in a vehicle, such as tire slip, electric powertrain dynamics, and regenerative braking. To address these challenges, Nonlinear Model Predictive Control (NMPC) improves MPC performances by incorporating nonlinear system models, allowing for more accurate control in real-world driving conditions.

This chapter introduces the NMPC formulation applied to the considered autonomous electric vehicles, describing its prediction model, cost function, constraints, and optimization approach. The objective of the NMPC controller is to guarantee safe and energy-efficient driving, while optimizing trajectory tracking, power consumption, and vehicle stability.

4.2 NMPC General Formulation

Nonlinear Model Predictive Control is an optimization-based control method that uses a simplified model of the system to predict future states and optimize control inputs based on a cost function and subject to several constraints.

In particular, NMPC works by solving an **optimization problem** at each control interval, where the objective is to find the optimal control inputs that minimize the **cost function** while satisfying system **constraints**.

The NMPC operates with a **moving horizon** approach, meaning that at each time step:

- The controller predicts the future vehicle states over a finite time horizon N, called **prediction horizon**, based on the current state and control inputs.;
- A constrained nonlinear optimization problem is solved to determine the optimal control sequence that minimizes a given cost function while respecting system constraints;
- Only the first control input of the optimized sequence is applied to the vehicle. This process, known as the **receding horizon** strategy, is repeated at each time step using updated sensor data.

By looking at it from a mathematical point of view, the NMPC seeks to solve, at each step, an optimization problem by minimizing the cost function subject to system dynamics and constraints. In the discrete-time domain, this can be expressed as:

$$\min J = \sum_{i=0}^{N-1} L\Big(x(i), u(i)\Big) + \phi\Big(x(N)\Big) \tag{4.1}$$

subject to:

$$x(i+1) = f\Big(x(i), u(i)\Big)\, g\Big(x(i), u(i)\Big) \tag{4.2}$$

where:

- J is the cost function;
- N is the prediction horizon;
- $x(i)$ represents the state of the system at instant i;
- $u(i)$ represents the control input at instant i;
- L is the stage cost function that quantifies the performance of the system at each time step;
- ϕ is the terminal cost function that captures the desired final state of the system;
- f represents the system dynamics, and describe how the state evolves over time as function the current state and control input;
- g represents the system constraints, which limit the feasible state and input space.

By iteratively solving this optimization problem, the NMPC algorithm generates a sequence of control inputs that minimizes the cost function while satisfying the dynamics and constraints of the system.

4.3 NMPC Development

Having established the general formulation of NMPC in the previous paragraph, this section formally defines the structure of the NMPC framework that had been developed during this thesis. Specifically, it details the prediction model employed, the cost function formulated to optimize performance, and the constraints imposed to ensure feasibility and compliance with physical limitations.

4.3.1 Prediction model for NMPC

In this work, the prediction model used for the NMPC framework, which is essential for predicting future states over the prediction horizon, has been developed following the discussion on vehicle longitudinal dynamics and electric powertrain functioning presented in Chapter 3.

The system model, which results from the combination of vehicle longitudinal dynamics and electric powertrain equations, is formulated in a **state-space representation**. The selected **state variables** of the system are:

- $x_1 = SOC \rightarrow$ Battery State of Charge (SOC);
- $x_2 = x \rightarrow$ Vehicle Position;
- $x_3 = \dot{x} \rightarrow$ Vehicle Velocity.

The **control input** of the state-space model is the electric motor torque, which is the output of the NMPC controller:

$$u = T_{EM} \tag{4.3}$$

To express the state-space equations, it is necessary to rewrite the equations from Chapter 3 in terms of the selected state variables and control input.

The longitudinal dynamics of the vehicle are governed by the forces opposing motion, which can be classified into three main components:

- Aerodynamic resistance (3.3) : $F_{\text{aero}} = \frac{1}{2}\rho A_f C_d x_3^2$;
- Rolling resistance (3.2): $F_{\text{roll}} = m\ g\ f_r \cos(\theta)$;
- Slope resistance (3.4): $F_{\text{slope}} = m\ g\ \sin\theta$.

The resulting longitudinal dynamic equation of the vehicle can be written as:

$$M_{\text{veh}}\, \dot{x}_3 = F_{\text{EM}} - F_{\text{roll}} - F_{\text{aero}} - F_{\text{slope}} \tag{4.4}$$

where M_{veh} is the equivalent mass of the vehicle, $\dot{x}_3$ is the vehicle acceleration and F_{EM} is the traction force provided by the powertrain.

The electric motor converts electrical energy into mechanical torque, and its rotational speed is related to the vehicle velocity (x_3) by the gearbox ratio:

$$\omega_{\text{EM}} = \frac{\tau_{\text{gb}}\, x_3}{r_w} \tag{4.5}$$

where ω_{EM} is the motor angular velocity, τ_{gb} is the gearbox ratio, and r_w is the wheel radius.

The mechanical power delivered by the motor is given by:

$$P_{\text{EM}} = \omega_{\text{EM}}\, T_{\text{EM}} = \omega_{\text{EM}}\, u \tag{4.6}$$

where P_{EM} represents the power output of the motor.

The battery model is essential for tracking the State of Charge (SOC) and ensuring optimal energy management. The power supplied by the battery is computed as:

$$P_b = \frac{P_{\text{EM}}}{(\eta_{\text{EM}}\, \eta_{\text{inv}})^{\text{sign}(P_{\text{EM}})}} \tag{4.7}$$

where η_{EM} and η_{inv} are the efficiencies of the electric motor and inverter, respectively.

The open-circuit voltage and the internal resistance of the battery are functions of the SOC:

$$V_{\text{OC}} = N_s\, V(x_1) \tag{4.8}$$

$$R_{\text{O}} = \frac{N_s}{N_p}\, R(x_1) \tag{4.9}$$

Where N_s is the number of series-connected battery cells, N_p is the number of parallel-connected cells, V and R are the function of the open circuit voltage and resistance of the battery depending on the SOC that have been obtained from experimental data.

Finally, the battery current is calculated as:

$$I_b = \frac{V_{\text{OC}} - \sqrt{V_{\text{OC}}^2 - 4R_{\text{O}}P_b}}{2R_{\text{O}}} \tag{4.10}$$

By combining all the previous equations, the **state equation** of the system can be obtained:

$$\dot{x}_1 = f(x_1, x_3, u) = -\frac{I_b}{Q_{\text{nom}}\,\eta_{\text{batt}}^{\text{sign}(I_b)}} \tag{4.11}$$

$$\dot{x}_2 = f(x_3) = x_3 \tag{4.12}$$

$$\dot{x}_3 = f(x_3, u) = \frac{\left(\frac{\eta_{\text{gb}}\,\tau_{\text{gb}}}{r_w}\right) T_{\text{EM}} - F_{\text{roll}} - F_{\text{aero}}}{M_{\text{veh}}} \tag{4.13}$$

So, the state derivative vector is calculated:

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} -\frac{I_b}{Q_{\text{nom}}\eta_{\text{batt}}}\text{sign}(I_b) \\ x_3 \\ \frac{(\eta_{\text{gb}}/\phi)T_{\text{EM}} - F_{\text{roll}} - F_{\text{aero}}}{M_{\text{veh}}} \end{bmatrix} \tag{4.14}$$

And the output vector is given:

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \tag{4.15}$$

4.3.2 Prediction Horizon

In the Nonlinear Model Predictive Control (NMPC) framework, the prediction horizon plays a crucial role in determining the controller's ability to anticipate future states and optimize control actions accordingly. The prediction horizon is defined by two key parameters: the sampling time T_s and the number of prediction steps N.

The sampling time T_s represents the time interval at which the NMPC updates its predictions and computes the optimal control inputs. A smaller T_s allows for more frequent updates, improving accuracy, but at the cost of higher computational demand. Conversely, a larger T_s reduces computational burden but may lead to delayed responses to dynamic changes.

The prediction horizon length N defines how far into the future the controller predicts the system's behavior. A longer horizon enables better foresight in decision-making, but it also increases the computational complexity of the optimization problem. Choosing N requires a trade-off between control accuracy and real-time feasibility.

For this thesis, the NMPC controller was designed with a sampling time of $T_s = 0.5$ seconds and a prediction horizon of $N = 5$ steps. These values were chosen to ensure a balance between control responsiveness and computational efficiency, allowing the system to react effectively to changes in the environment while maintaining real-time feasibility for implementation.

As function of the values T_s and N, the value of the prediction period T_p can be obtained:

$$T_p = T_s \cdot N = 2.5 \ s \tag{4.16}$$

4.3.3 Cost Function

The cost function in a Nonlinear Model Predictive Control (NMPC) framework serves as the optimization criterion that guides the control decisions over the prediction horizon. It is designed to balance multiple objectives, such as minimizing energy consumption, ensuring safe inter-vehicle distances, maintaining optimal speed, and preserving battery State of Charge (SOC).

The cost function introduced in the NMPC formulation can be expressed as:

$$J = \lambda_1 \|P_{\text{EM}}\|^2 + \lambda_2 \|\text{SOC} - \text{SOC}_{\text{ref}}\|^2 + \lambda_3 \|d - d_{\max}\|^2 + \lambda_4 \|v - v_{\text{ref}}\|^2 + \lambda_5 \|a_k - a_{k-1}\|^2 \tag{4.17}$$

where:

- P_{EM} is Electric Motor Power;

- SOC is Battery State of Charge;

- SOC_{ref} is the reference Battery SOC;

- d is Vehicle distance to preceding vehicle;

- $d_{\max}$ is Maximum allowable distance;

- v is Vehicle speed;

- v_{ref} is Reference speed;

- a_k is the vehicle acceleration at instant k;

- a_{k-1} is the vehicle acceleration at instant k-1;

- λ_i are the weighting coefficients for cost function tuning.

Each component of the cost function is designed to address a specific **control objective**:

- $\lambda_1 \|P_{\text{EM}}\|^2$ minimizes the electric power used by the motor to improve efficiency and extend vehicle range;
- $\lambda_2 \|\text{SOC} - \text{SOC}_{\text{ref}}\|^2$ keeps the battery SOC close to a reference value to prevent excessive discharge;
- $\lambda_3 \|d - d_{\max}\|^2$ prevents the vehicle from falling too far behind in adaptive cruise control (ACC) scenarios;
- $\lambda_4 \|v - v_{\text{ref}}\|^2$ maintains the desired velocity while respecting traffic conditions ensuring smooth acceleration and braking.
- $\lambda_5 \|a_k - a_{k-1}\|^2$ aims to reduce the variation in terms of acceleration between two consecutive instants, improving the level of comfort perceived by passengers.

The cost function weights (λ_1, λ_2, λ_3, λ_4, λ_5) play a fundamental role in shaping the behavior of the NMPC controller, as they determine the relative importance of different optimization objectives, such as energy efficiency, tracking accuracy, and driving comfort. The tuning of these weights was carried out through an iterative process, balancing performance trade-offs to achieve optimal control behavior.

The final values of the cost function weights, as determined from the tuning process, are summarized in Table 4.1 below.

Weight	Cost Parameter	Value
λ_1	P_{EM}	1
λ_2	SOC	1000
λ_3	d	$8 \cdot 10^5$
λ_4	v	100
λ_5	a	500

Table 4.1: Cost function weights values

Summarizing, the NMPC cost function optimizes the vehicle's behavior by balancing energy efficiency, safety, and trajectory tracking. Each term plays a crucial role in ensuring the vehicle maintains optimal performance, operates efficiently, and adheres to safety constraints. The weighting factors λ_i allow for fine-tuning based on specific driving conditions and priorities.

4.3.4 Constraints

In a Nonlinear Model Predictive Control (NMPC) framework, constraints play a crucial role in ensuring that the system operates within physical, safety, and operational limits. In the NMPC developed in this thesis, the constraints in can be categorized into:

- System Dynamics Constraints (ensuring consistency with vehicle equations).
- State Constraints (e.g., SOC, position, speed).
- Control Input Constraints (e.g., motor torque limits).
- Safety Constraints (e.g., minimum safe following distance).

System Dynamics Constraints

The system dynamics are enforced using discretization constraints to ensure that the predicted states respect the equations of motion:

$$x_{k+1} = x_k + T_s f(x_k, u_k) \tag{4.18}$$

in the code part:

```
st_next = X(:,k+1);
f_value = f(st,con);
st_next_euler = st + (Ts * f_value);
g = [g; st_next - st_next_euler]; % compute constraints
```

```
...
args.lbg(1:3*(N+1)) = 0;
args.ubg(1:3*(N+1)) = 0;
```

This means that the predicted next state x_{k+1} must follow the system dynamics using Euler discretization. The equality constraint ensures that NMPC predictions are consistent with the actual system model. The lower bound (lbg) and upper bound (ubg) are set to zero, ensuring the system equations hold exactly.

State Variables Constraints

The state variables are physically constrained to ensure realistic behavior:

- Battery State of Charge (SOC):

$$\mathrm{SOC}_{\min} \leq \mathrm{SOC}_k \leq \mathrm{SOC}_{\max} \tag{4.19}$$

  The SOC of the battery cannot fall below a critical threshold (to prevent deep discharge), while the upper limit prevents overcharging during regenerative braking.

- Position Constraints:

$$0 \leq x_2 \leq \infty \tag{4.20}$$

  The vehicle position is constrained to be nonnegative (ensuring it does not move backward), while no upper bound is imposed, allowing unrestricted forward motion.

- Speed Limits:

$$v_{\min} \leq v_k \leq v_{\max} \tag{4.21}$$

  The vehicle must operate within a defined speed range to comply with the safety and legal limits of traffic.

Control Input Constraints

The value of the control output, which corresponds to the electric motor torque, is constrained to prevent excessive power demand and ensure the feasibility of real components operations:

$$T_{\mathrm{EM},\min} \leq T_{\mathrm{EM},k} \leq T_{\mathrm{EM},\max} \tag{4.22}$$

The lower bound prevents negative torque beyond regenerative braking capabilities, and the upper bound ensures the motor does not exceed its maximum power output.

Safety Constraints

A constraint is imposed to maintain a safe distance from the preceding vehicle:

$$d_{\text{veh}}(k) - d_{\text{ref}}(k) \geq d_{\text{safe}}(v_k) \tag{4.23}$$

in the code part:

```
for k=1:N
    speed = X(3,k);
    g = [g; P((n_states + n_controls) * k + 1) - X(2,k) -
   Safety_Distance_function(speed)];
end
...
args.lbg(3*(N+1)+1:3*(N+1)+N) = 0;
args.ubg(3*(N+1)+1:3*(N+1)+N) = inf;
```

The lower bound (lbg) of this constraint is zero, which means that the vehicle must not violate the minimum distance limit, while the upper bound (ubg) is set to infinity, meaning there is no restriction on exceeding the safe distance.

The function *Safety_Distance_function(speed)* dynamically calculates the minimum safety distance required for a vehicle to maintain a safe gap from the vehicle in front, based on its current speed:

$$d_{\text{safety}} = \frac{1.5 \cdot v_{\text{veh}}^2}{2 \cdot a_{\text{brake}}} + t_{\text{reaction}} \cdot v_{\text{veh}} \tag{4.24}$$

where:

- d_{safety} is the computed safety distance;
- v_{veh} is the vehicle velocity (m/s);
- a_{brake} is the maximum deceleration (m/s^2);
- t_{reaction} is the reaction time (s),

The function takes into account two main factors: the braking distance required to bring the vehicle to a complete stop in case of an emergency (given by the first term) and the reaction time of the driver (or controller) $t_{reaction}$.

In summary, NMPC constraints ensure that the vehicle respects physical, safety, and operational limits, allowing the controller to optimize energy efficiency, trajectory tracking, and vehicle stability without exceeding the system capabilities.

Chapter 5

Implementation and Simulation

5.1 Implementation of the NMPC Framework

The NMPC algorithm was implemented in MATLAB, using CasADi, an open-source symbolic framework for automatic differentiation and nonlinear optimization.

MATLAB was chosen due to its significant support for numerical computing, vehicle modeling, and control system simulation, while CasADi was utilized to efficiently formulate and solve the NMPC optimization problem.

5.1.1 Introduction to CasADi

CasADi is a powerful tool for symbolic and algorithmic differentiation, widely used in optimization-based control applications, including NMPC. It provides an efficient interface for defining nonlinear systems, cost functions, constraints, and their gradients, significantly reducing computational complexity.

In this work, CasADi is also exploited for discretizing the system dynamics, as the NMPC formulation introduced in the previous chapter 4.1 is based on a continuous-time model. The discretization process ensures that the vehicle dynamics can be expressed in a suitable form for real-time optimization, enabling the NMPC controller to make accurate predictions and optimize control actions over a finite prediction horizon.

Key advantages of using CasADi in this work:

- **Symbolic Computation:** CasADi allows defining optimization variables, cost functions, and constraints **symbolically**, ensuring efficient computation of gradients and Hessians.

- **Discretization of System Dynamics**: Since NMPC is formulated in continuous time, CasADi is used to apply discretization methods (such as Euler or Runge-Kutta integration), transforming the continuous-time vehicle model into a discrete-time representation suitable for real-time optimization. This process ensures that the NMPC framework can predict the system's evolution at discrete time steps, maintaining both accuracy and computational efficiency.
- **Integration with MATLAB:** It seamlessly integrates with MATLAB, allowing direct implementation of NMPC for real-time control applications.
- **Solver Compatibility:** CasADi provides access to nonlinear optimization solvers such as IPOPT, which are essential for handling constrained NMPC problems.
- **Computational Efficiency:** By leveraging CasADi's automatic differentiation, the NMPC controller benefits from fast and accurate derivative calculations, improving real-time feasibility.

5.2 Constant Time Gap (CTG) Controller

To estimate the performance improvements achieved by NMPC, particularly in terms of battery energy efficiency and driving safety, the proposed NMPC controller is compared with a Constant-Time-Gap (CTG) controller under the same simulation conditions. The aim of this comparison is to quantify the benefits of predictive control strategies over traditional rule-based approaches.

The Constant-Time-Gap (CTG) controller is a widely used rule-based strategy for implementing an Adaptive Cruise Control (ACC) technology. It is based on the idea of adjusting the vehicle acceleration/deceleration based on the distance and on the relative speed between the controlled vehicle and the preceiding vehicle according to a predefined equation [19]:

$$a_{ego} = -\frac{1}{h}\left(\lambda\ \delta_{ego} + \dot{\varepsilon}_{ego}\right) \tag{5.1}$$

where:

- $\varepsilon_{ego} = x_{ego} - x_{leading}$ is the relative distance;
- $\dot{\varepsilon}_{ego} = \dot{x}_{ego} - \dot{x}_{leading}$ is the relative velocity;
- h is the desired time gap;
- $L_{des} = h\ \dot{x}_{ego}$ is desired space distance;

- $\delta_{ego} = \varepsilon + L_{des}$ is the spacing error;
- λ is a tuning parameter;
- a_{ego} is the calculated vehicle acceleration.

In this thesis, the CTG controller has been developed on Simulink, and applied to a vehicle model totally corresponding to the vehicle model that had been presented in the previous chapter.

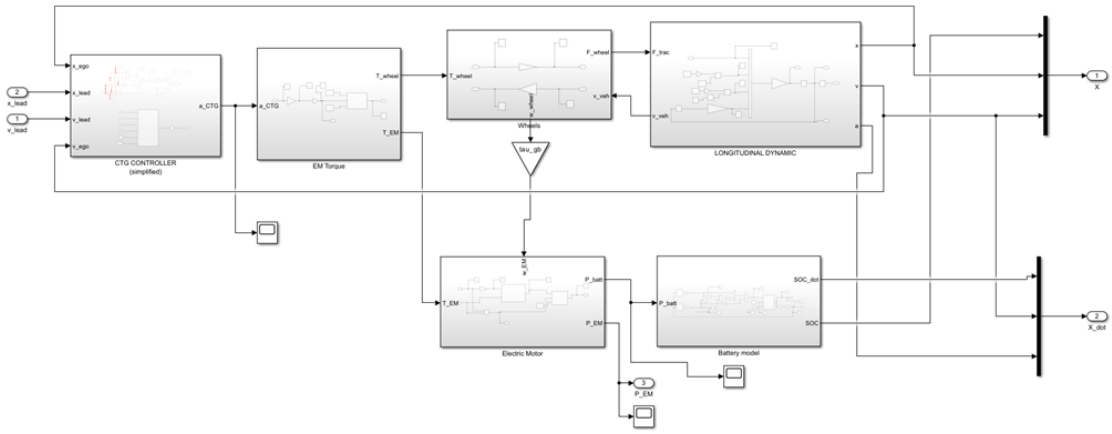

Figure 5.1: Controlled vehicle model

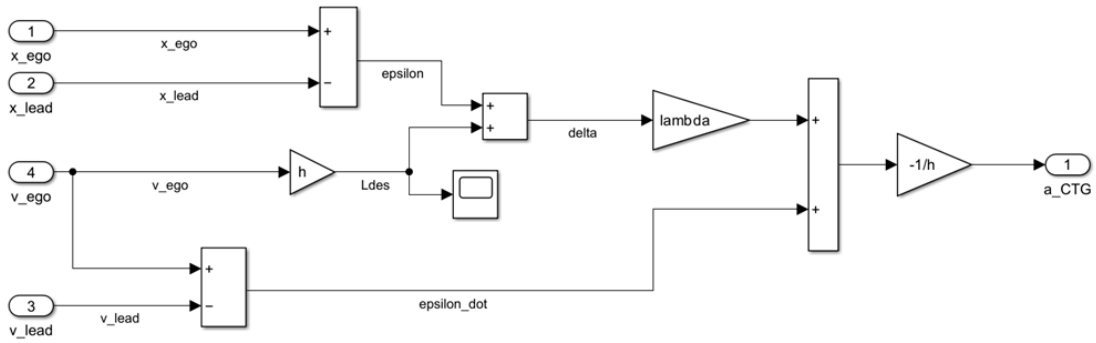

Figure 5.2: CTG controller

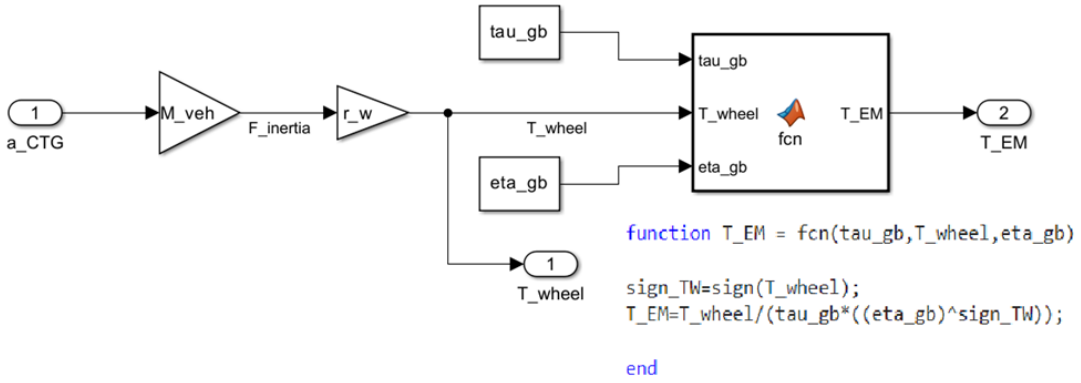

Figure 5.3: EM Torque block

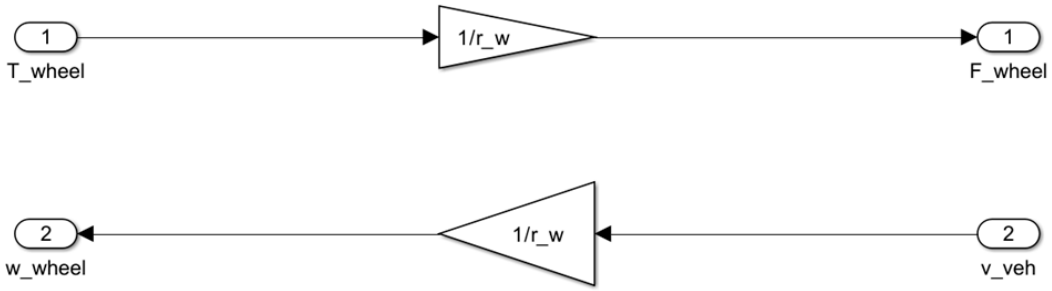

Figure 5.4: Wheel model block

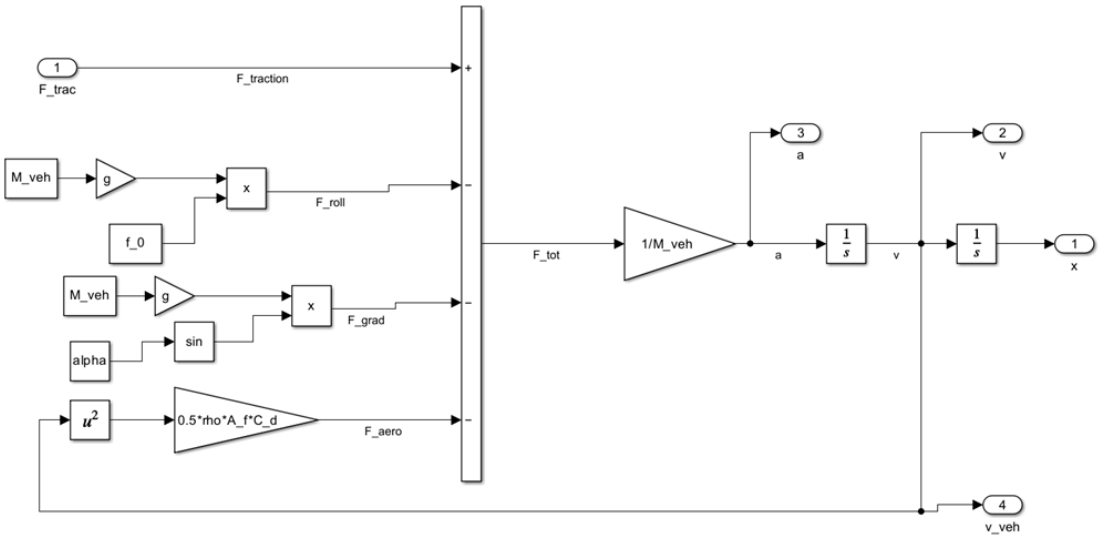

Figure 5.5: Longitudinal dynamic block

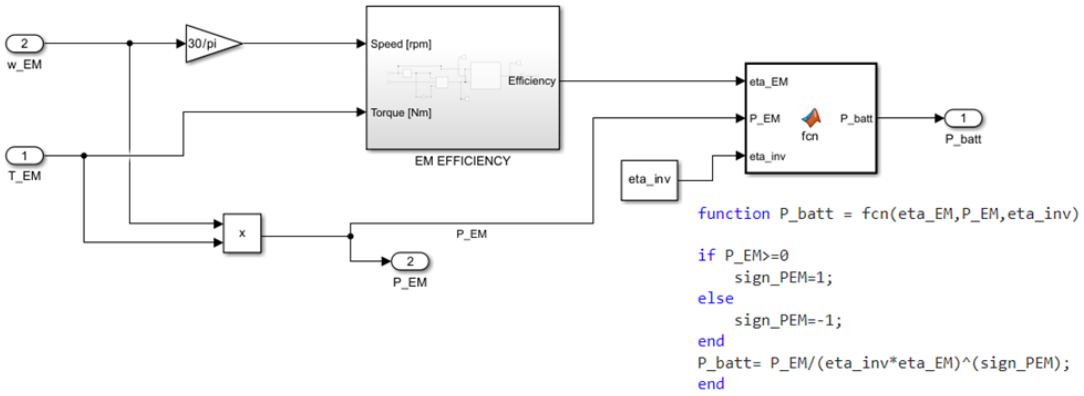

Figure 5.6: Electric Motor block

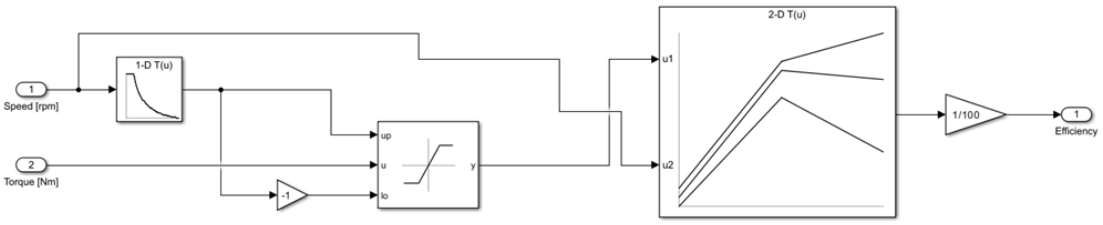

Figure 5.7: Electric Motor Efficiency block

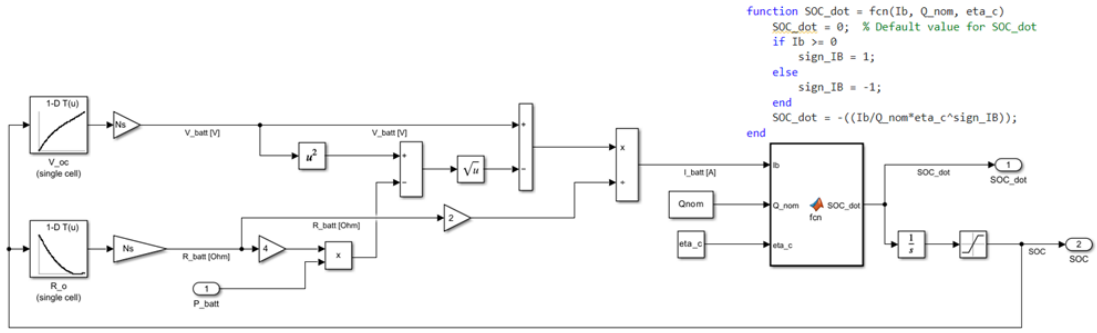

Figure 5.8: Battery Model block

The CTG controller adjusts acceleration and/or deceleration to ensure that the controlled vehicle maintains a predefined time gap [20]. However, since CTG does not optimize power usage or account for future predictions, it can be considered as a benchmark to investigate NMPC performance.

5.3 Simulations

This section describes the simulation scenarios designed to evaluate the performance of the NMPC controller in a realistic Adaptive Cruise Control (ACC) task. The goal of these scenarios is to test the ability of the NMPC controller to track a leading vehicle's position and speed while optimizing energy efficiency and maintaining a safe inter-vehicle distance.

5.3.1 Single-Vehicle ACC task

The first simulation scenario has been designed such that the vehicle controlled by the NMPC or by the CTG has to perform an Adaptive Cruise Control (ACC) task by following a leading vehicle while pursuing different objectives:

- **Ensure safe following distance:** The vehicle must never exceed the minimum safety distance from the leading vehicle;
- **Minimize energy consumption:** By optimizing acceleration and braking, the controller reduces battery energy usage;
- **Maximize comfort:** The control inputs should be applied smoothly, avoiding abrupt changes in acceleration.

The leading vehicle travels following the WLTP cycle in Fig.5.9, that is a standardized driving profile designed to simulate real-world driving conditions. It consists of multiple phases with varying speeds, accelerations, and decelerations, simulating urban, extra-urban and highway driving scenarios.

5.3.2 Platoon Simulation scenario

In addition to the Single-Vehicle Adaptive Cruise Control (ACC) task, a more complex Platoon simulation scenario was designed to evaluate the performance of the NMPC and CTG controllers in coordinating multiple vehicles' motion.

In this setup, a leading vehicle follows the predefined WLTP driving cycle, while four additional vehicles, each one controlled by an independent NMPC or CTG controller, follow the vehicle ahead in a structured platoon formation. Each vehicle regulates its motion based on the state of the vehicle directly in front.

An example of a platoon architecture, in this case developed on Simulink to evaluate CTG performances on platoon, is reported in Fig.5.10.

The objective of this scenario is to ensure **platoon stability**, where speed deviations and distance fluctuations decrease as they propagate down the platoon, leading to a smooth, energy-efficient and synchronized motion of all vehicles.

Platoon Stability Objective

The main goal of this scenario is to achieve **platoon stability**, that is defined as:

$$\lim_{t\to\infty} \|v_i(t) - v_{i-1}(t)\| \to 0, \quad \forall i \in \{2, \dots, 5\} \tag{5.2}$$

where:

- $v_i(t)$ is the velocity of the i-th vehicle at time t;

- $v_{i-1}(t)$ is the velocity of the preceiding vehicle.

The platoon stability condition ensures that speed deviations across the platoon converge to zero over time, preventing propagation or amplification of disturbances.

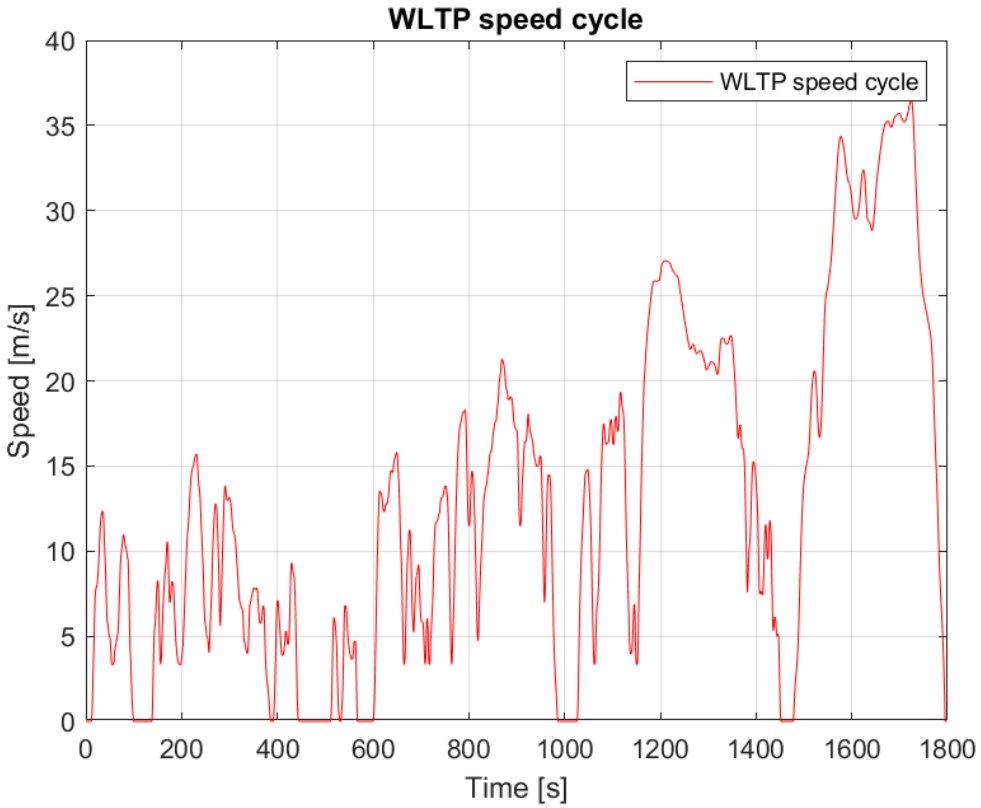

Figure 5.9: WLTP speed cycle

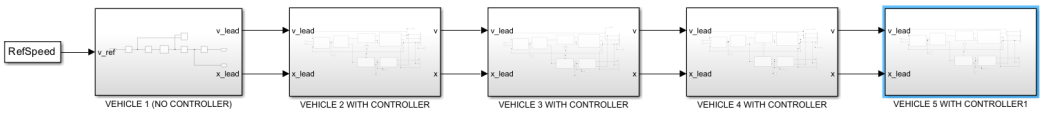

Figure 5.10: Architecture of the platoon developed to test CTG controller

Chapter 6

Simulation results

This chapter presents and analyzes the results obtained from the NMPC simulations, evaluating its performance in two key scenarios: the **single-vehicle Adaptive Cruise Control (ACC) task** and the **platoon simulation scenario**.

The objective is to evaluate the effectiveness of NMPC in optimizing vehicle energy consumption, while ensuring speed tracking, maintaining safe inter-vehicle distances and enhancing driving comfort.

A comparative analysis with the Constant Time Gap (CTG) controller is also conducted to highlight the advantages of developed controller over traditional rule-based approaches.

6.1 Single-Vehicle ACC simulation results

In the first simulation scenario, the NMPC and CTG controllers were tested in an Adaptive Cruise Control (ACC) task, where the controlled vehicle was required to follow a leading vehicle executing the WLTP driving cycle.

In the following plots, the results of CTG and NMPC simulations are reported in terms of:

- State of Charge (SOC) of the controlled vehicle;
- Position of the two vehicles;
- Speed of the two vehicles;
- Acceleration of the two vehicles;
- Torque provided by the Electric Motor;
- Power spent by the Electric Powertrain during the simulation.

6.1.1 State of Charge (SOC)

The two plots in Fig.6.1 illustrate the evolution of the battery State of Charge (SOC) over time for a vehicle controlled by NMPC (top plot) and CTG (bottom plot) during the simulation. The blue line represents the reference SOC (set for bothe the controller to 0.8), while the red line shows the SOC evolution of the controlled vehicle in each case. This comparison highlights the advantages of the predictive control (NMPC) over a traditional rule-based approach (CTG) in terms of energy efficiency.

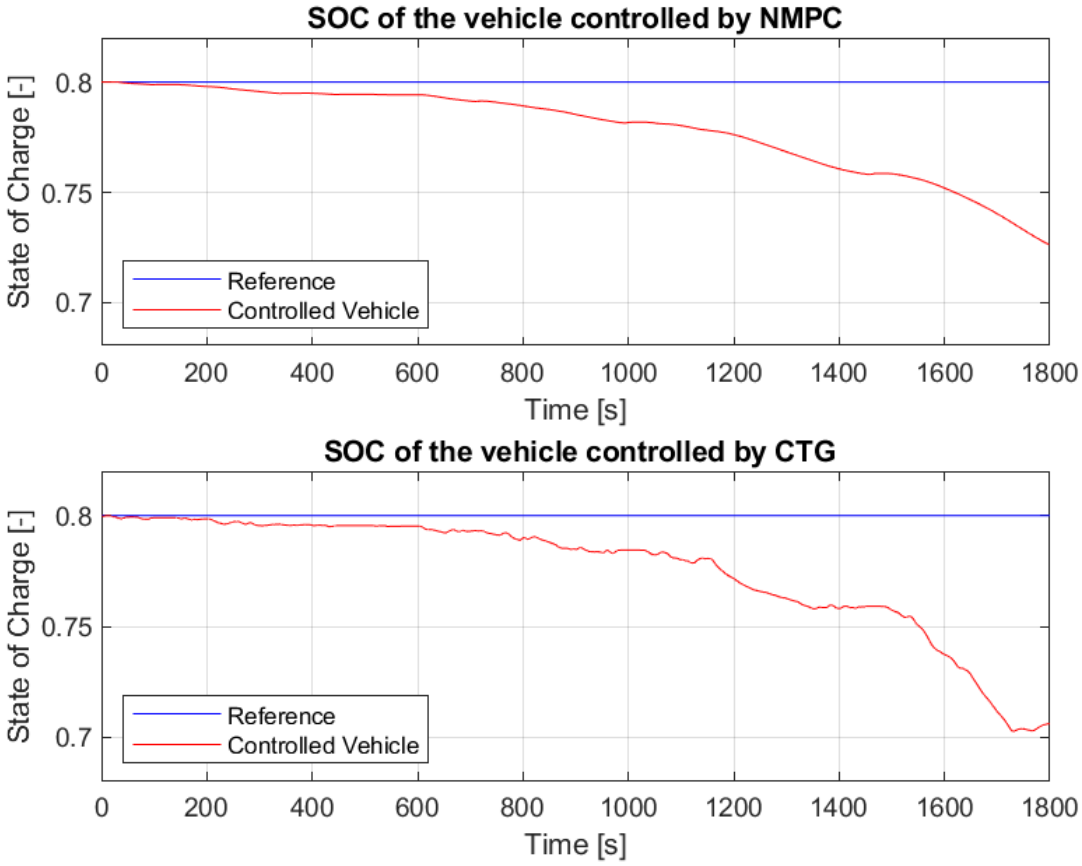

Figure 6.1: Comparison of SOC evolution for NMPC and CTG

As it can be seen, The SOC of the vehicle controlled by NMPC (top plot) declines gradually and smoothly, indicating a more efficient energy management strategy. In contrast, the CTG-controlled vehicle (bottom plot) experiences a more pronounced SOC drop, particularly after 1400 seconds, suggesting higher energy consumption and lower efficiency. Moreover, for what concerns SOC fluctuations,

it is evident that the SOC curve in the CTG plot shows higher oscillations, which is due to more frequent and sudden accelerations and brakings. In contrast, NMPC regulates energy flow more smoothly, resulting in a more consistent SOC decline. This is another big advantage of NMPC controller with respect to CTG controller, since more frequent SOC oscillations may results in a significant deterioration of battery health.

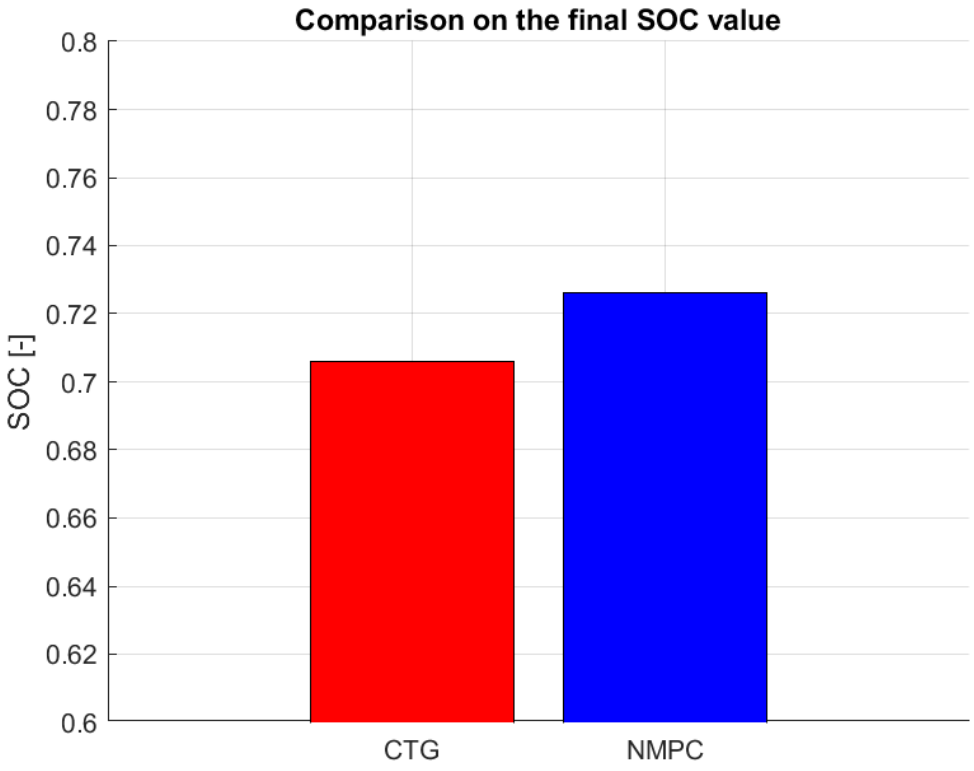

Figure 6.2: SOC value at the end of the simulation for CTG and NMPC controllers

The Fig.6.2 instead represents a comparison of the final values of the SOC of the two vehicles at the end of the simulations. As it can be seen, the NMPC control strategy allows the vehicle to mantain an higher value of SOC (around 72.6%) at the end of the simulation with respect to CTG (almost 71%). This is a further confirmation of the ability of the NMPC to improve the vehicle's range.

6.1.2 Positions

The two plots in Fig.6.3 illustrate the position evolution of the controlled vehicle compared to the leading vehicle over time, for both NMPC (top plot) and CTG (bottom plot). These results highlight how well each control strategy maintains a proper following behavior, ensuring the controlled vehicle stays close to the leader while respecting safety constraints.

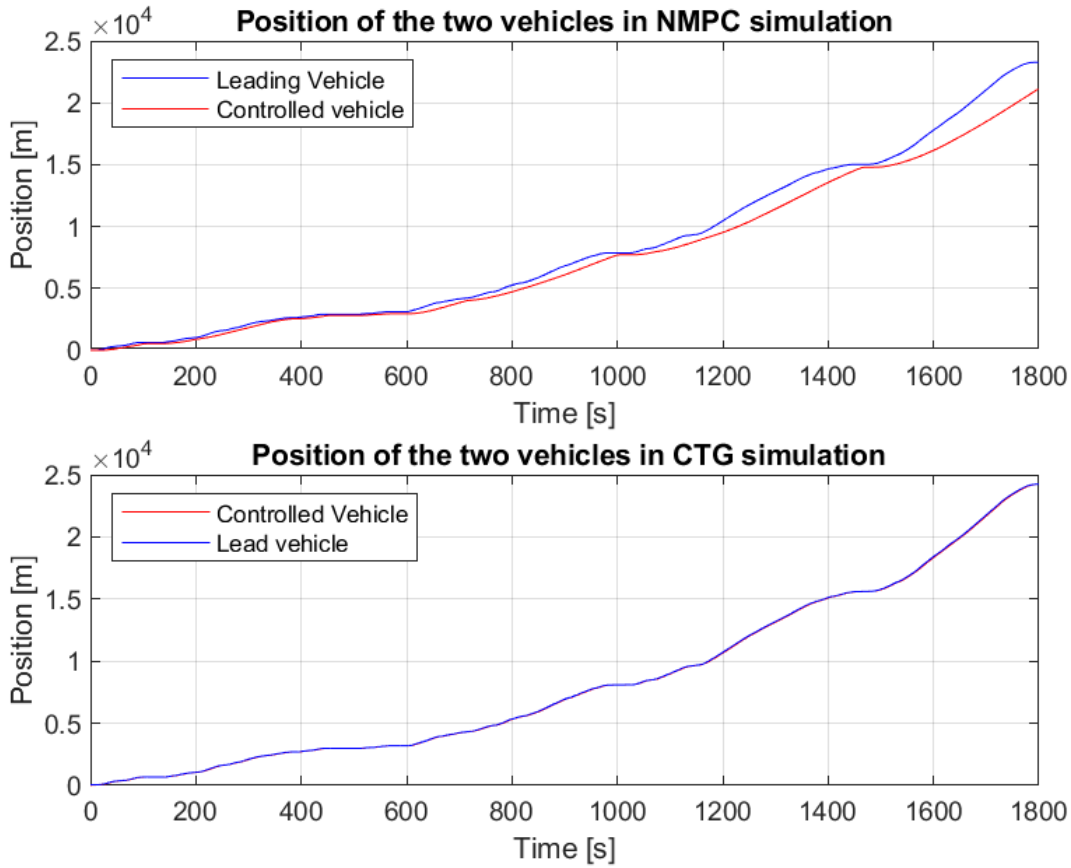

Figure 6.3: Comparison of position plot for NMPC and CTG simulations

As it can be seen, the CTG controlled vehicle has an higher tracking behavior with respect to NMPC, but since it strictly follows every variation in the speed of the leading vehicle, this can results in more frequent accelerations and decelerations lowering the driving comfort experienced by passengers.

A drawback of NMPC, as seen in the figure above, is that its focus on energy efficiency causes the vehicle to gradually increase its distance from the leader. Although this reduces energy consumption, it may conflict with the range limitations

of current LiDAR, radar, and camera sensors, which require consistent tracking distances for reliable perception. This highlights the need for a balance between energy efficiency and sensor constraints in real-world implementations.

6.1.3 Speed

The two plots in Fig.6.4 compare the speed evolution of the controlled vehicle in both NMPC (top plot) and CTG (bottom plot) simulations. These graphs illustrate how well each controller tracks the speed of the leading vehicle and adjust its velocity accordingly.

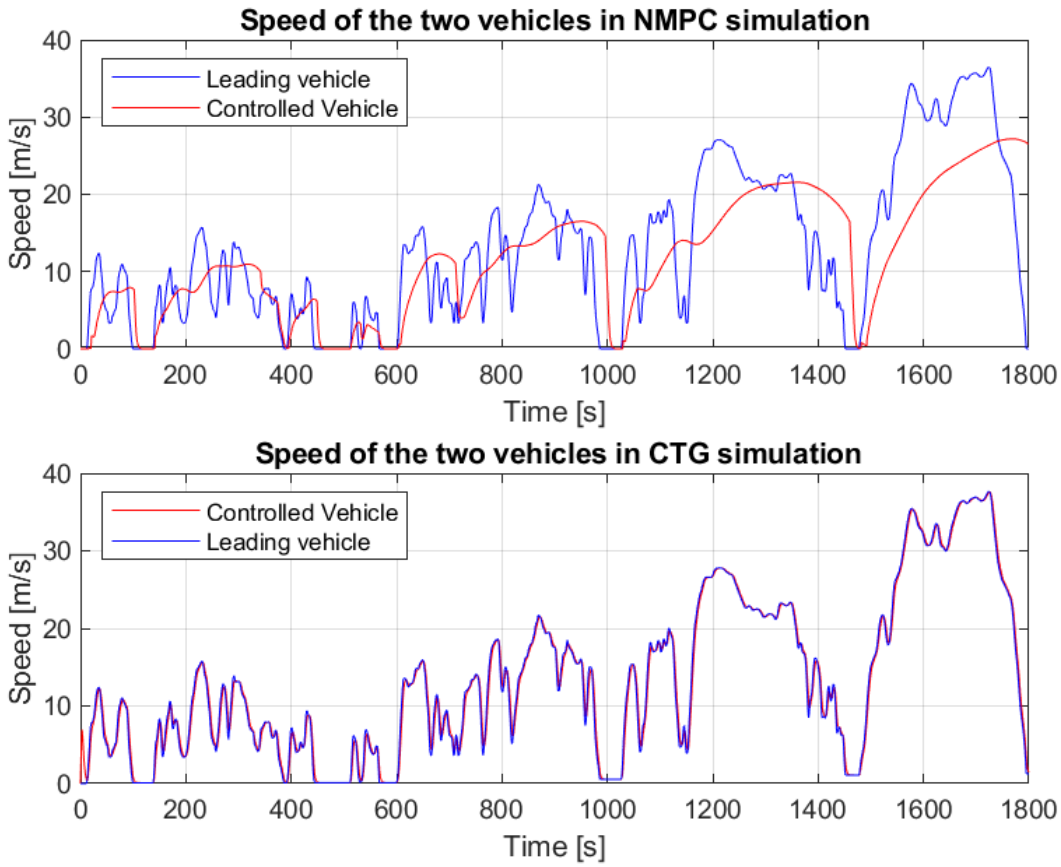

Figure 6.4: Comparison of speed plot for NMPC and CTG simulations

As it can be seen, the NMPC-controlled vehicle (red line) follows the leading vehicle (blue line) with a more gradual and controlled speed profile. In fact, the predictive nature of NMPC allows it to anticipate upcoming speed changes, leading to fewer abrupt accelerations and decelerations compared to CTG.

In practice, the NMPC controller smooths out the vehicle's velocity profile, leading to a more comfortable and energy-efficient driving experience. The CTG-controlled vehicle, instead, shows higher speed oscillations, which can lead to increased energy consumption and unnecessary damage on the powertrain.

6.1.4 Acceleration

The two graphs in Fig.6.5 illustrate the acceleration profiles of the controlled vehicle compared to the leading vehicle over time for both NMPC (top plot) and CTG (bottom plot). These graphs provide insights into how smoothly each controller manages acceleration and braking, which directly impacts energy efficiency, driving comfort, and safety.

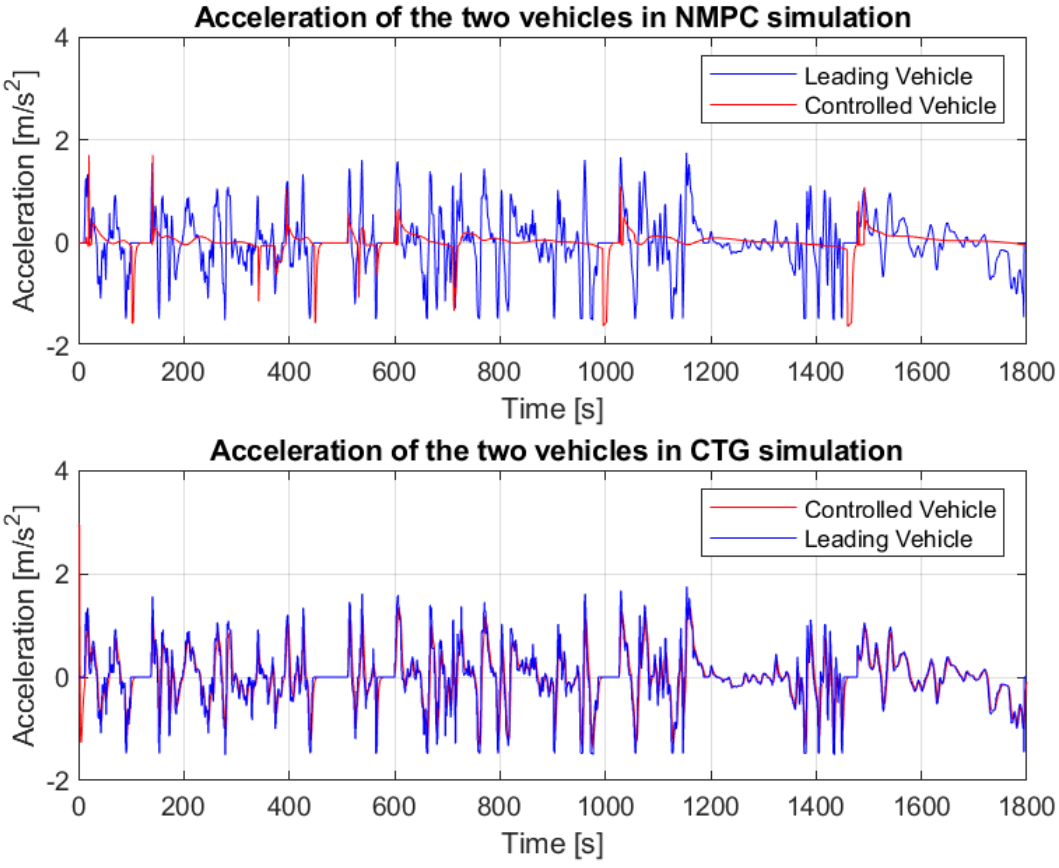

Figure 6.5: Comparison of acceleration plot for NMPC and CTG simulations

In the NMPC simulation, the controlled vehicle's acceleration (red line) remains more stable and less oscillatory than the reference acceleration (blue line). The NMPC controller effectively filters out excessive acceleration variations, resulting in a more comfortable and energy-efficient drive. There are fewer fluctuations, meaning that NMPC minimizes effectively unnecessary acceleration and braking inputs.

In the CTG simulation, the controlled vehicle acceleration (red line) shows more oscillations and higher spikes. Since CTG is a reactive controller, it struggles to soften rapid variations in acceleration, leading to an increase in jerk. The frequent spikes indicate that CTG responds to each change in the leading vehicle speed, resulting in more rapid acceleration and braking events.

6.1.5 Electric Motor Torque

The two graphs in Fig.6.6 compare the electric motor (EM) torque demand for the controlled vehicle in both NMPC (top plot) and CTG (bottom plot) simulations. These graphs illustrate how each controller regulates motor torque to optimize vehicle dynamics, energy consumption, and driving smoothness.

In the NMPC simulation, the commanded torque (red line) closely follows the reference torque (blue line) with a less oscillatory behavior. NMPC applies a smoother torque, avoiding excessive fluctuations, which results in more energy-efficient and comfortable driving. In fact, torque variations are reduced over time, especially after 1000s, demonstrating that NMPC optimizes power delivery as the vehicle stabilizes.

On the other hand, the CTG simulation exhibits a torque profile (red line) with significant oscillations and abrupt changes.The higher variability indicates that CTG reacts aggressively to speed changes of leading vehicle, causing frequent and unnecessary torque adjustments. In fact, since CTG does not anticipate future speed variations, it compensates reactively, leading to higher energy consumption and lower efficiency.

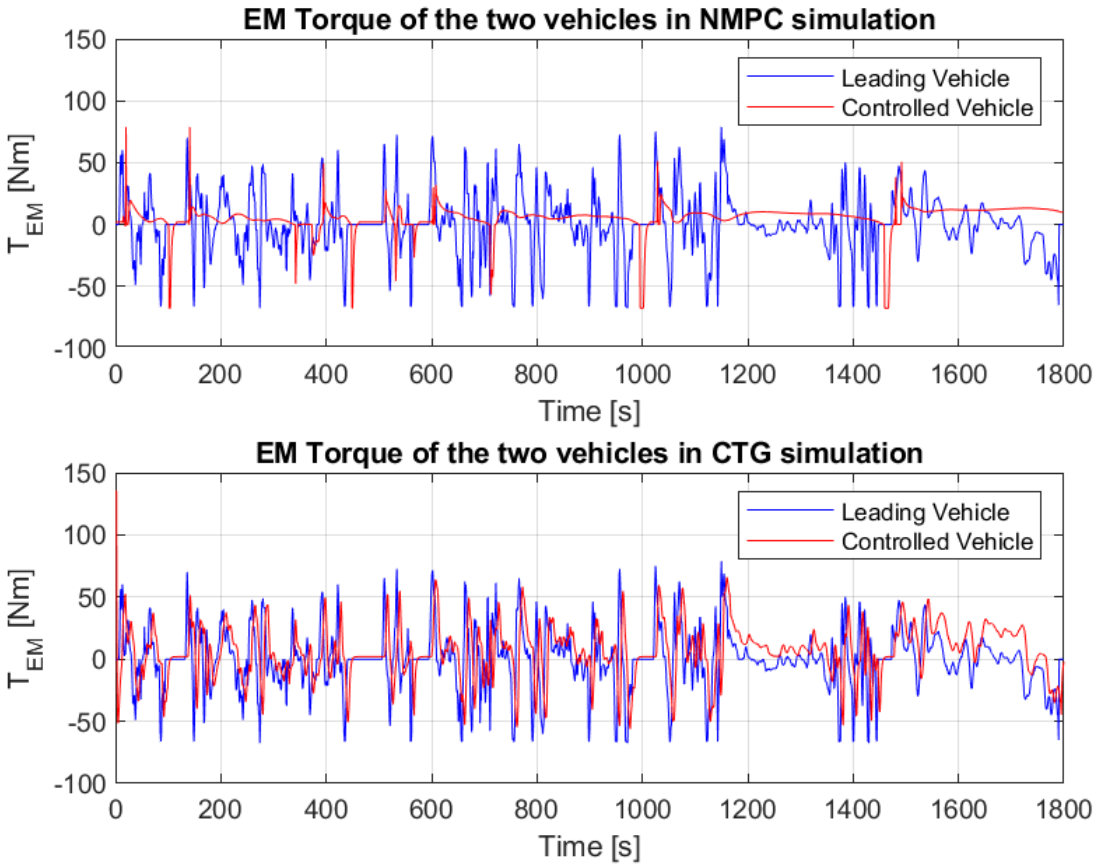

Figure 6.6: Comparison of EM Torque plot for NMPC and CTG simulations

6.1.6 Electric Motor Power

The two plots in Fig.6.7 represent a comparison on the electric motor (EM) power usage for the controlled vehicle under NMPC (top plot) and CTG (bottom plot) during the simulation. These results illustrate how efficiently each controller manages power distribution, acceleration demands, and regenerative braking.

In the NMPC simulation, the power demand (red line) exhibits less variability and smoother transitions compared to the leading vehicle power profile (blue line).This means that NMPC efficiently regulates power delivery, avoiding excessive peaks, which results in better energy optimization and reduced power waste.Also, regenerative braking events (negative power values) are more pronounced and stable, meaning NMPC maximizes energy recuperation during deceleration.

In the CTG simulation, the EM power demand fluctuates significantly, with high

peaks and abrupt variations. Unlike NMPC, CTG does not optimize power transitions effectively, leading to unnecessary accelerations and inefficient regenerative braking.

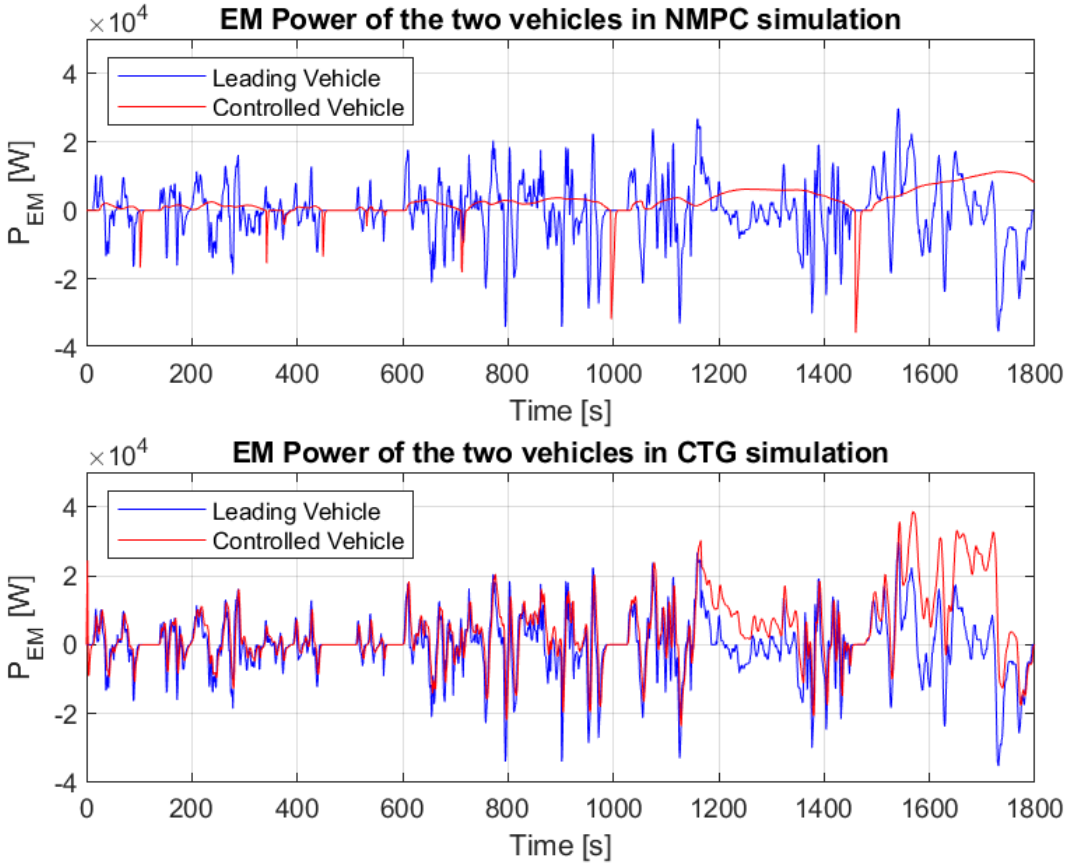

Figure 6.7: Comparison of EM Power plot for NMPC and CTG simulations

Lastly, the Fig. 6.8 reports a comparison on the total value of power that the Electric Motor provides to the vehicle during the whole simulation for CTG (red) and NMPC (blue) controllers. It is evident that the value of EM power provided in case of CTG controller is higher than for NMPC simulation and this provides further confirmation that the NMPC effectively improves the energy efficiency of the vehicle.

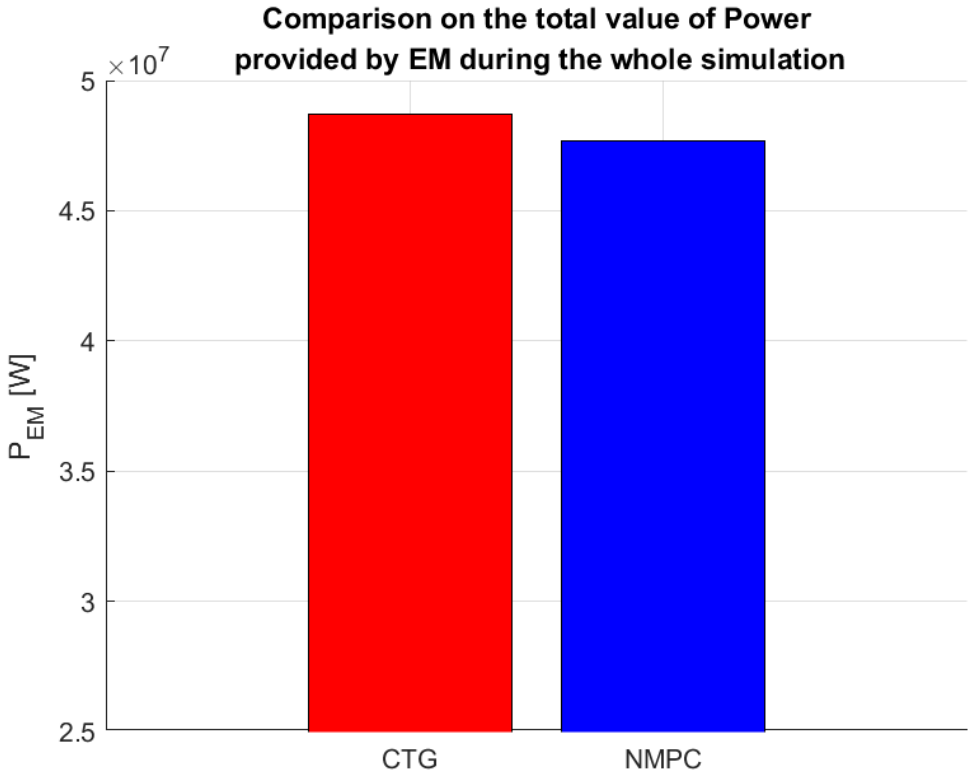

Figure 6.8: Total value of power provided by the EM over the simulation for CTG and NMPC controllers

6.2 Platoon simulation results

The second simulation scenario aimed to evaluate NMPC's ability to control a platoon of four vehicles, each following a leading vehicle executing the WLTP cycle. In these simulations, the objective, in addition to safety and energy saving goals already addressed previously, was to achieve platoon stability, ensuring that speed deviations and spacing fluctuations diminish as they propagate through the platoon.

6.2.1 State of Charge (SOC)

The two plots in Fig. 6.9 compare the State of Charge (SOC) evolution of the vehicles in the platoon controlled by NMPC (top plot) and CTG (bottom plot)

controllers. These graphs illustrate how each control strategy affects the energy efficiency across multiple vehicles, showing the battery charge consumption trend over time.

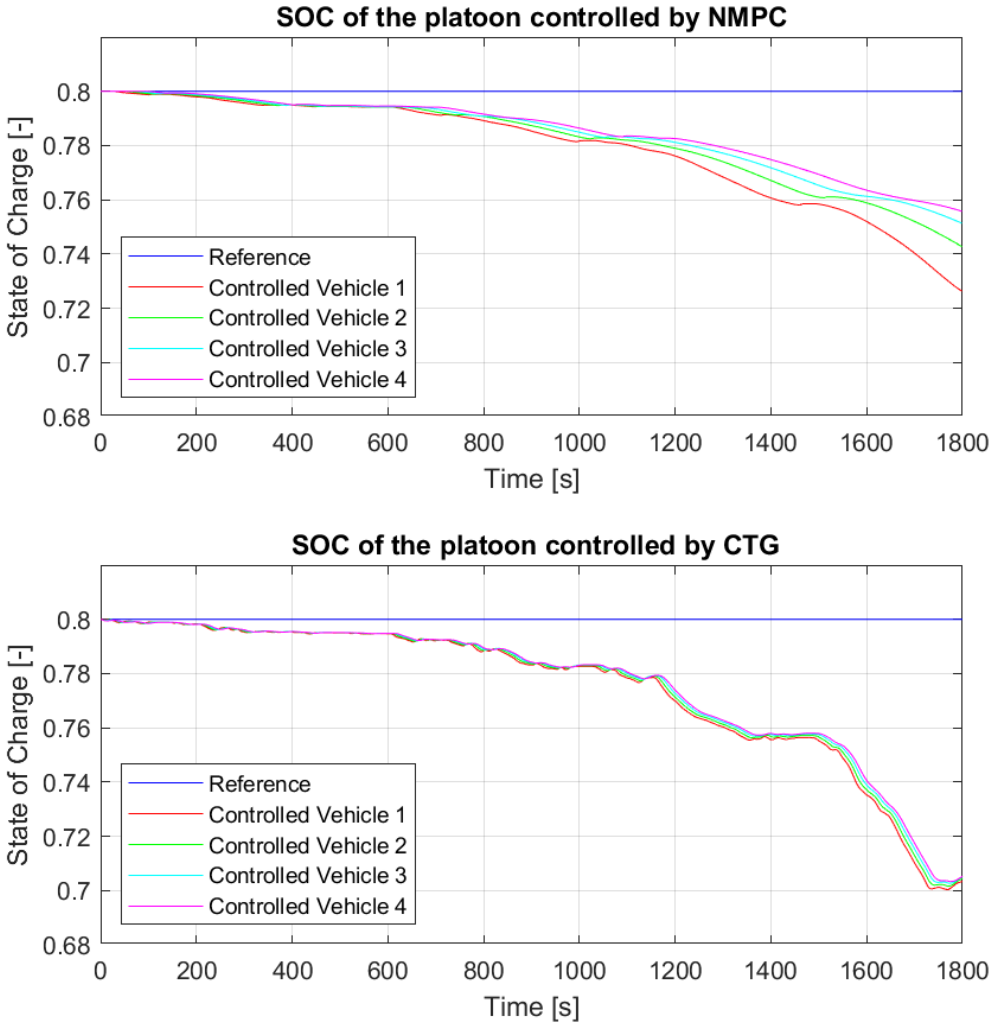

Figure 6.9: Comparison on SOC evolution over the platoon for NMPC and CTG simulations

In the NMPC-controlled platoon (top plot), the SOC of the vehicles declines gradually and consistently, with minimal deviations between vehicles, suggesting that NMPC optimizes power usage.

Unlike NMPC, the CTG-controlled vehicles experience larger SOC deviations,

which suggests that the reactive control approach causes greater variations in energy consumption between different vehicles.

By looking at the deviations of the SOC curves through the multiple vehicles in the platoon, it is evident that while in the CTG-controlled platoon the curves are really close to each other, resulting in a similar value of SOC at the end of the simulation, in NMPC-controlled platoon the curves are more distant from one another, resulting in higher values of residuale SOC at the end of the simulation for vehicles at the end of the platoon.

The difference in terms of battery charge consumption curves represents the ability of the NMPC to effectively filter the fluctuations in the power usage of the WLTP cycle when passing from a vehicle to another in the platoon.

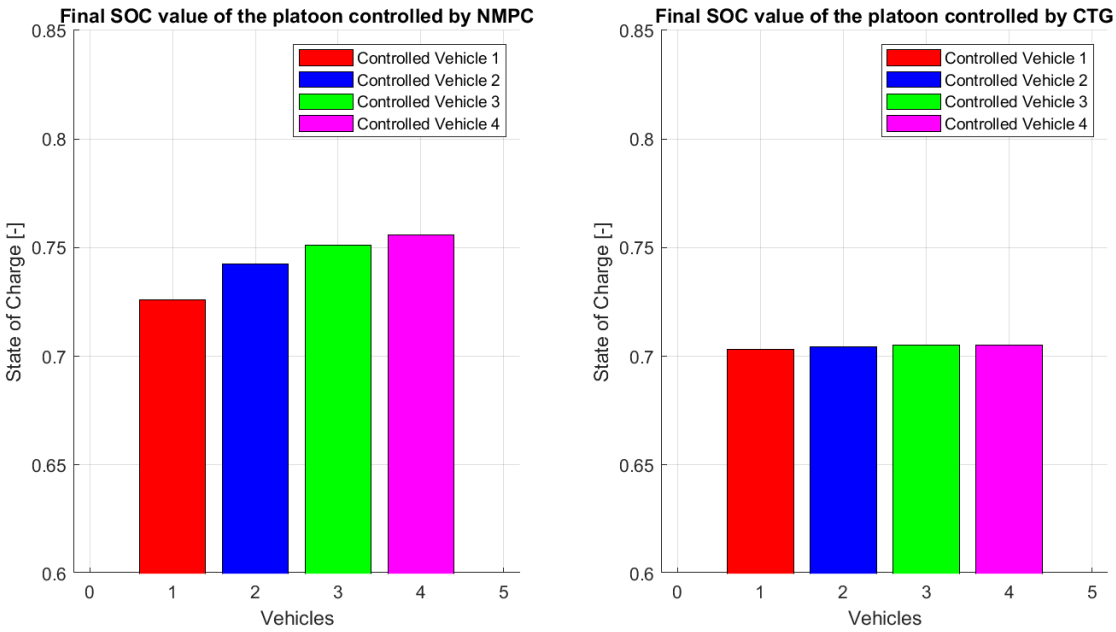

Figure 6.10: Comparison on final value of SOC at the end of the simulation for NMPC and CTG platoons

This is also confirmed by the Fig. 6.10 in which the residual values of battery SOC at the end of the simulation are reported for each vehicle in the platoon. It is evident that while in the CTG-controlled platoon the residual values of SOC are really similar for each vehicle, the NMPC effectively provides an improvement in the energy efficiency when passing from a vehicle to the next.

6.2.2 Positions

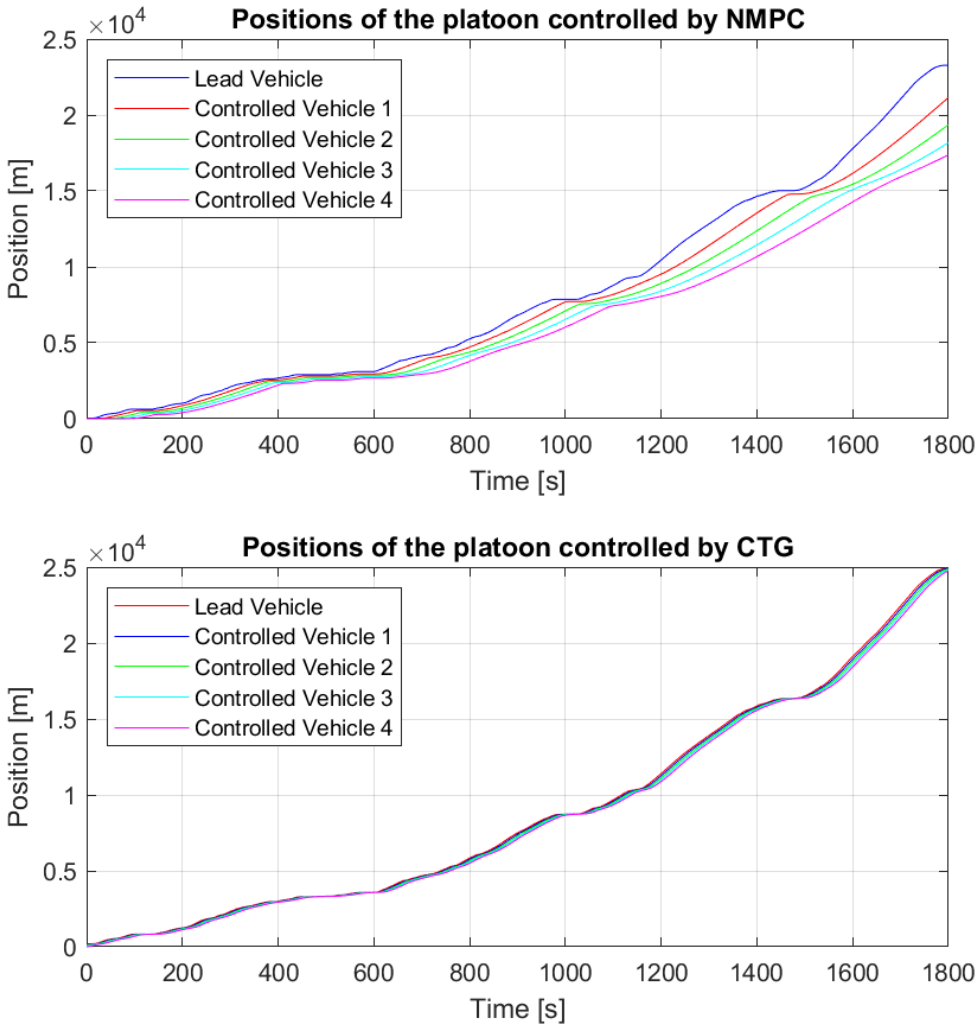

Figure 6.11: Comparison on position evolution over the platoon for NMPC and CTG simulations

In the Fig. 6.11 a comparison of the positions of the platoons controlled by NMPC (top plot) and CTG (bottom plot) is reported. This comparison highlights significant differences in how inter-vehicle distances and overall formation stability are managed.

In the NMPC-controlled platoon, the vehicles maintain progressively increasing

distances from the leading vehicle, but in a structured and coherent manner. The gradual separation between the curves suggests that the NMPC controller dynamically regulates each vehicle's speed, increasing the value of safety distance at higher velocities. This behavior is fundamental to ensure soft driving and efficient energy consumption, reducing the propagation of disturbances along the platoon.

In contrast, the CTG-controlled platoon exhibits a much more rigid and synchronized behavior, with the position curves of all vehicles almost overlapping. This indicates that all vehicles follow the leading vehicle maintaining a fixed time gap without flexibly adapting to speed variations. While this approach ensures a consistent following distance, it may be less energy-efficient and less responsive to sudden changes in traffic conditions. Furthermore, the lack of adaptability can lead to sudden braking or dangerous maneuvers, especially when the leader's speed fluctuates significantly.

6.2.3 Speed

The Fig. 6.12 represents a comparative analysis on the speed evolution of each vehicle in the NMPC (top plot) and CTG (bottom plot) controlled platoons. The comparison of vehicle speeds aims to highlight the differences in how each control strategy manages speed regulation and platoon stability.

In the NMPC-controlled platoon, the speeds of the individual vehicles show a more flexible and adaptive response to the variations in the leading vehicle's velocity. The vehicles gradually adjust their speeds, with an evident smoothing effect that prevents sudden accelerations and decelerations. This behavior suggests that NMPC is effectively optimizing each vehicle's speed trajectory while maintaining a balance between tracking performance, energy efficiency and safety.

In contrast, the CTG-controlled platoon shows a much more synchronized and rigid speed response. All vehicles closely follow the lead vehicle's speed variations, with their trajectories nearly overlapping during the simulation. While this ensures a consistent following behavior, it also suggests that CTG enforces a fixed time gap without smoothly adapting to changes in acceleration. As a result, the vehicles are more prone to reactive braking and acceleration maneuver, which can lead to higher energy consumption, potential powertrain stress and increased perceived discomfort. The lack of adaptability in CTG can also contribute to a higher risk of traffic-induced oscillations, where fluctuations in the leading vehicle speed may propagate throughout the platoon, causing less efficient energy usage and potential safety concerns.

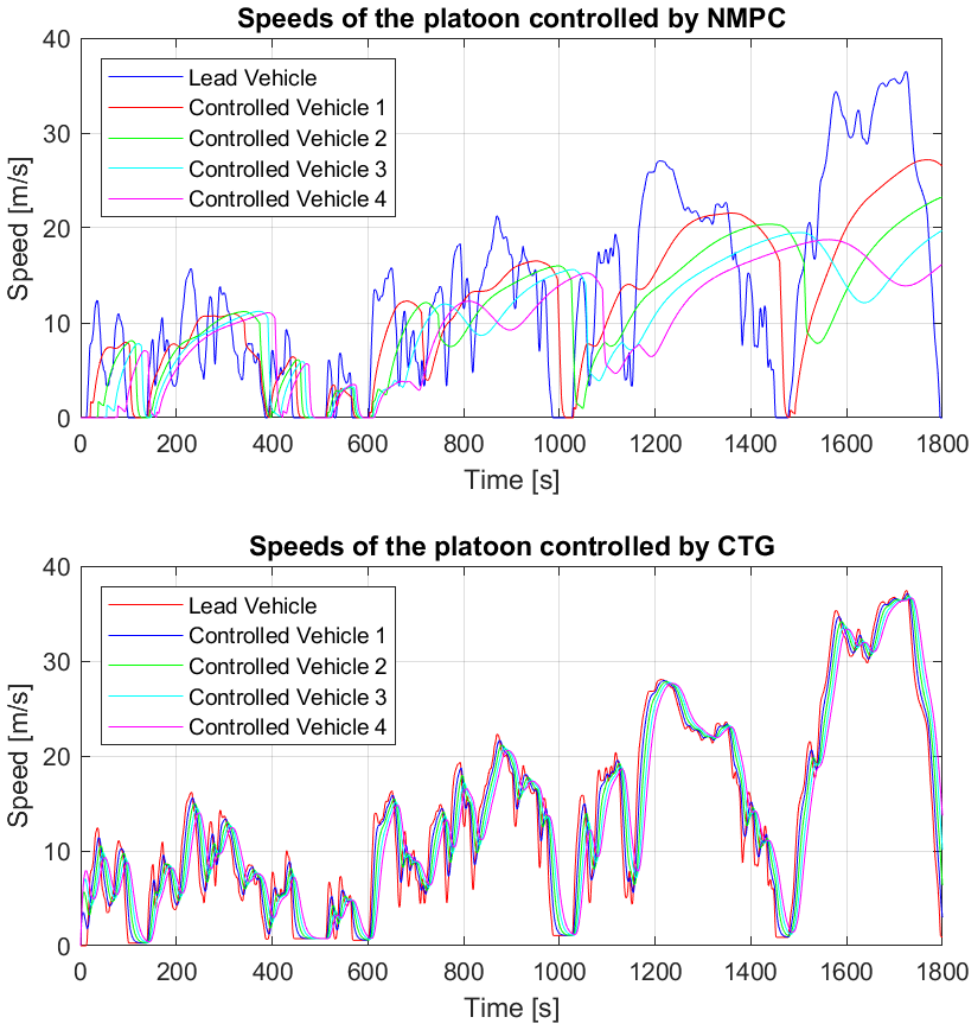

Figure 6.12: Comparison on speed evolution over the platoon for NMPC and CTG simulations

6.2.4 Acceleration

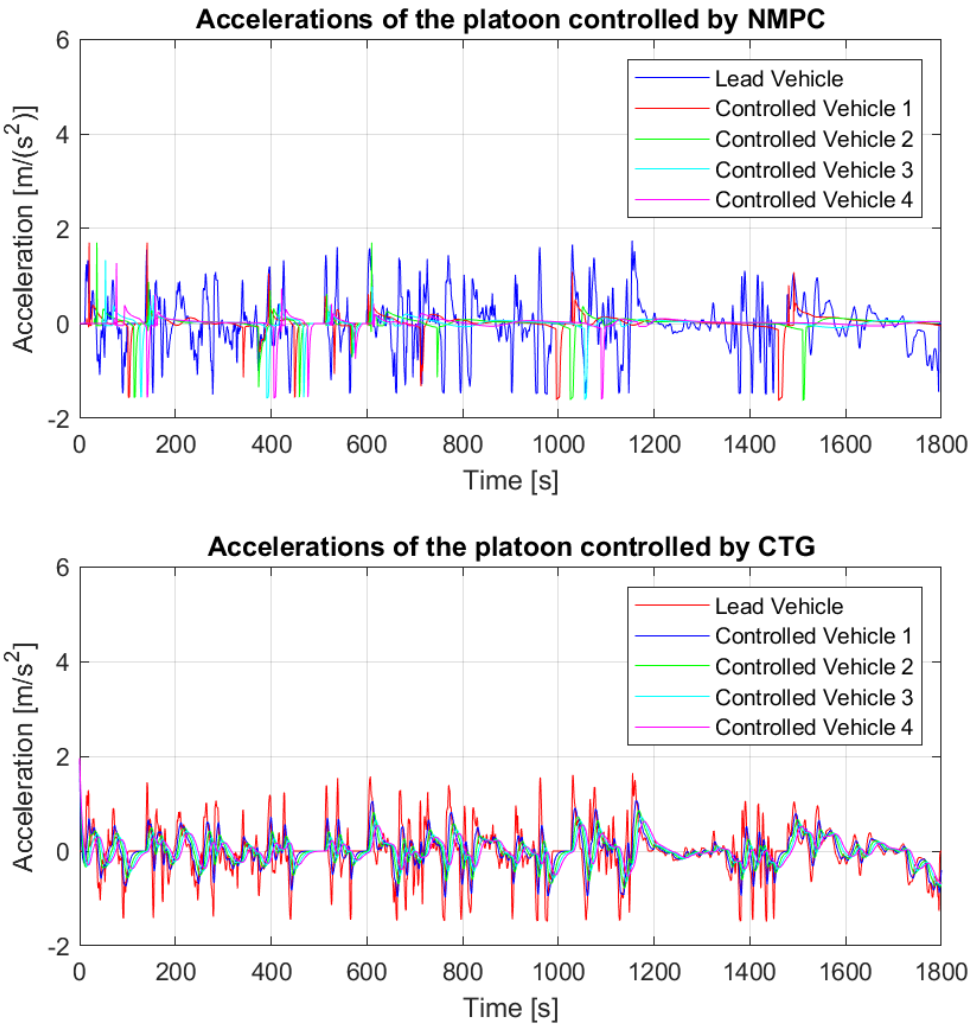

Figure 6.13: Comparison on acceleration evolution over the platoon for NMPC and CTG simulations

The Fig. 6.13 describe the evolution in terms of acceleration of the vehicle in the platoon controlled by NMPC (top plot) and CTG (bottom plot).

The analysis of acceleration profiles in the NMPC- and CTG-controlled platoons highlights how each control strategy manages to handle vehicle dynamics, particularly in terms of smoothness and stability, resulting in driving comfort perceived by passengers.

In the NMPC-controlled platoon, the acceleration curves of the controlled vehicles show a softened response to the fluctuations of the leading vehicle, indicating that NMPC effectively mitigates variations and prevents excessive oscillations from propagating through the platoon.

The vehicles maintain more stable and controlled acceleration patterns, with fewer spikes, allowing for reducing the presence of aggressive accelerations or brakings and optimizing the energy usage for each vehicle in the platoon.

On the contrary, the CTG-controlled platoon displays a high synchronization between vehicles in terms of acceleration, the acceleration curves almost perfectly follows the lead vehicle's fluctuations, indicating a more reactive but less adaptive control approach.

This lack of filtering means that any acceleration oscillation from the leading vehicle is immediately transferred throughout the platoon, leading to higher energy consumption, increased stress of powertrain component, and a less comfortable ride. The frequent peaks suggest that CTG depend on constant corrections rather than on predictive adjustments, resulting in potentially higher power losses due to unnecessary acceleration maneuvers.

6.2.5 Electric Motor Torque

The Fig.6.14 illustrate the plot of the Electric Motor Torque throughout the simuation for each vehicle in the platoons controlled respectively by the NMPC (top plot) and CTG (bottom plot). The comparison of those two plots is helpful to catch informations on how each control strategy manages to regulate the torque command to reach a trade-off between the objectives of trajectory tracking, energy efficiency and comfort.

In the NMPC-controlled platoon, the controlled vehicles demonstrate a significantly smoother and controlled torque response with respect to the highly oscillatory torque of the leading vehicle. NMPC efficiently filters out fluctuations, allowing the vehicles to maintain a stable torque distribution that minimizes sudden changes. This results in reduced vehicle powertrain stress and improved energy efficiency, as the system avoids excessive torque variations that could lead to unnecessary energy expenditure or damage of components. As the simulation advances, NMPC stabilizes the torque profiles, indicating that the system is successfully adapting to driving conditions while maintaining efficient energy management.

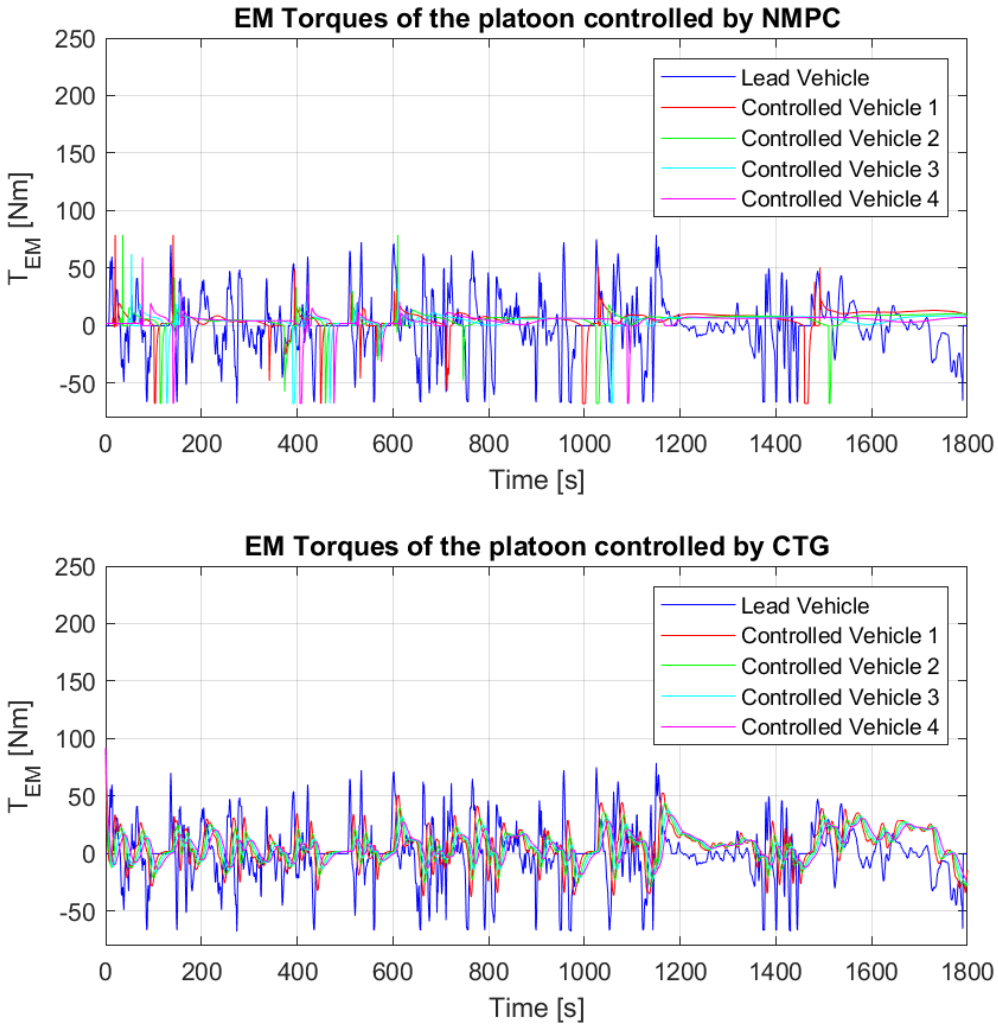

Figure 6.14: Comparison on EM Torque evolution over the platoon for NMPC and CTG simulations

By contrast, in the CTG-controlled platoon the torque profiles of the controlled vehicles closely follow one another, mirroring the leading vehicle's variations almost perfectly. While this may seem advantageous in terms of maintaining consistent inter-vehicle distances, it also amplifies fluctuations instead of mitigating them, leading to a more rigid and less adaptive driving strategy. The lack of smoothing in CTG leads to a propagation of disturbances from the lead vehicle directly through the platoon, increasing the risk of unnecessary energy consumption and mechanical stress. Unlike NMPC, CTG does not optimize torque transitions dynamically,

which can result in higher power losses and a less comfortable driving experience.

6.2.6 Electric Motor Power

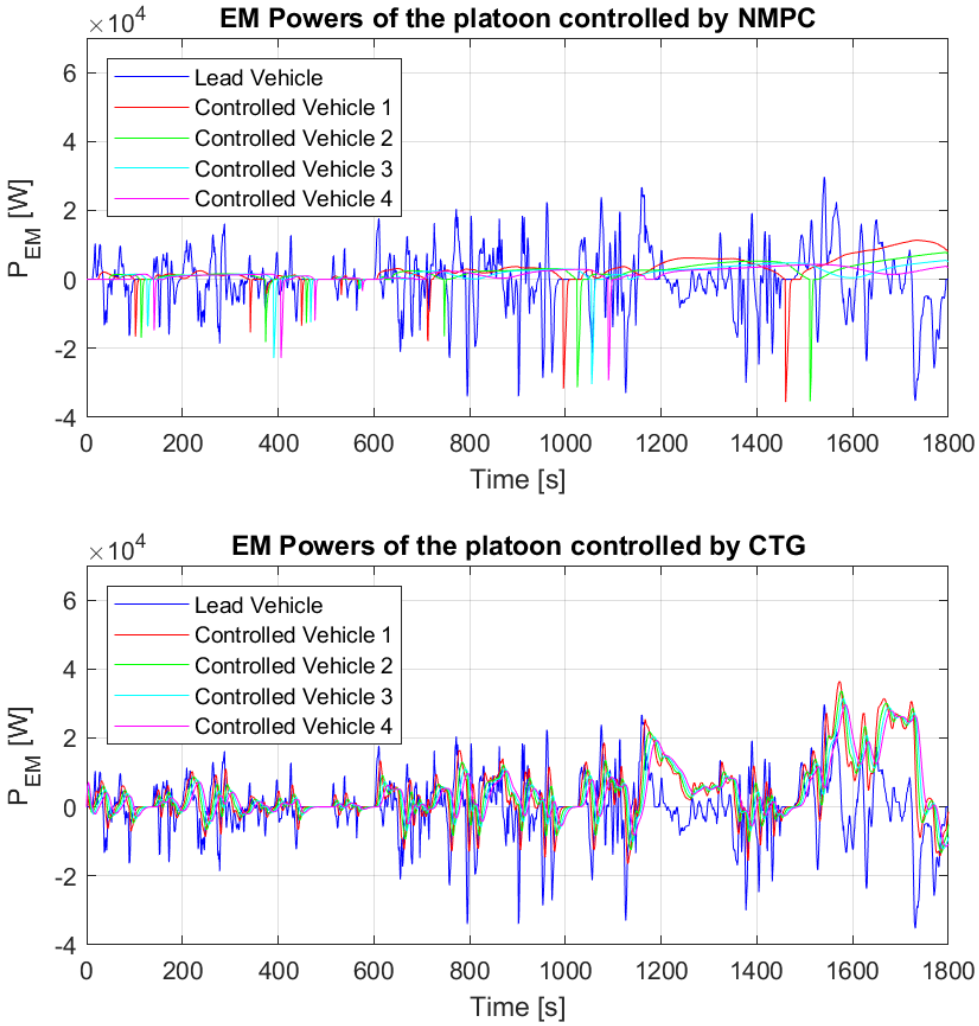

Figure 6.15: Comparison on EM Power evolution over the platoon for NMPC and CTG simulations

By looking at the Electric Motor (EM) power consumption in the NMPC- (top plot) and CTG-controlled (bottom plot) platoons in Fig. 6.15, it is clear that the two control strategies handle the energy usage in very different ways.

The power demand of the controlled vehicles in the NMPC platoon stays relatively stable and well-managed, despite the chaotic fluctuations of the leading vehicle. NMPC seems to be efficiently filtering out unnecessary variations, ensuring that each vehicle does not blindly copy every power rise or drop from the leading vehicle. Instead, the power output is more balanced, and we can see that regenerative braking is effectively utilized, as indicated by the occasional dips into negative power values. This means that NMPC is not just controlling the vehicles but it is actually optimizing energy use, recovering power when possible rather than wasting it.

In contrast, the CTG-controlled platoon shows a much more synchronized behavior, where the power demand curves of all vehicles practically overlap. This tells us that CTG enforces a strictier following strategy, where each vehicle responds almost instantly to power changes without considering whether those changes are actually necessary. As a result, there are frequent and sharp power variations, which could mean higher energy consumption over time. Another noticeable difference is in the regenerative braking phases: they seem much less effective in the CTG scenario, as the power dips aren't as deep or as consistent as in the NMPC case. This suggests that energy recovery isn't being maximized, leading to more energy being lost rather than reused.

Chapter 7

Conclusions

This thesis focused on the development and implementation of a Nonlinear Model Predictive Control (NMPC) strategy for longitudinal vehicle control in the context of autonomous driving. The primary objective was to enhance energy efficiency, driving stability, and safety compared to traditional rule-based approaches, such as the Constant Time-Gap (CTG) controller. The study was motivated by the increasing need for sustainable and efficient transportation solutions, particularly in the perspective of the challenges associated with electric vehicle autonomy.

To achieve this goal, a detailed vehicle and powertrain model was developed including vehicle longitudinal dynamics and a simplified electric powertrain model.

NMPC was designed to optimize control actions in terms of input torque, by minimizing a cost function that considered parameters such as energy consumption, safety, and driving comfort.

In order to evaluate the NMPC performances with respect to traditional rule-based Constant Time Gap controller, the model was tested in two main simulation scenarios:

- Single-vehicle Adaptive Cruise Control (ACC) task, where NMPC was compared against CTG in tracking a leading vehicle following the standardized WLTP driving cycle;

- Multi-vehicle Platoon scenario, in which NMPC was evaluated in maintaining a stable and energy-efficient platoon, again in comparison with CTG.

7.1 Key Findings

The simulation results demonstrated that NMPC significantly outperformed CTG in multiple aspects:

- **Energy efficiency:** vehicles controlled by NMPC consistently retained higher final State of Charge (SOC) values, demonstrating lower overall energy consumption compared to CTG. This resulted from NMPC's ability to optimize power distribution and reduce unnecessary acceleration and braking.
- **Driving smoothness and comfort:** NMPC resulted in smoother acceleration and speed profiles, reducing high-frequency fluctuations that were prominent in CTG. This behavior is crucial for reaching one of the major goals of this thesis: improving passenger comfort and minimizing drivetrain stress.
- **Platoon stability:** In the multi-vehicle simulations, the NMPC effectively mitigated the propagation of oscillations through the platoon, leading to better coordination between vehicles. Unlike CTG, which imposed a rigid following behavior, NMPC allowed for more adaptive spacing control, improving the overall safety and performance of the platoon.
- **Safety improvements:** One of the most significant advantages of NMPC over CTG is its ability to actively enforce safety constraints in real-time. The NMPC framework explicitly incorporated minimum safe following distances and speed constraints, ensuring that vehicles never violated safety margins. Unlike CTG, NMPC is able to predict and adjust to changes in the leading vehicle's behavior, reducing the likelihood of sudden braking events and unsafe proximity between vehicles. The improved stability in NMPC minimizes collision risks, making it a more reliable solution for autonomous driving in dynamic environments.

7.2 Next Steps

Given the excellent performance demonstrated by NMPC in terms of energy efficiency, driving stability, and safety, it can be considered a highly promising solution for future autonomous driving applications. The ability of NMPC to optimize power consumption, smooth out accelerations, and maintain stable platoon dynamics makes it a strong candidate for the implementation of next-generation transportation systems. However, to fully unlock its potential and ensure its

applicability in real-world applications, several areas require further investigation and improvement.

One of the most critical aspects to analyze is the **real-time implementation** of NMPC, since its computational complexity remains a challenge, particularly for embedded automotive systems with limited processing capabilities. Future studies could explore the optimization of numerical solvers, the use of parallel computing, or even the integration of learning-based approaches to fasten the prediction and optimization processes. Reducing computational load would make NMPC more suitable for real-world applications, in which fast decision-making is essential.

Another significant extension would be the **integration of lateral dynamics** into the control framework. While this study provided really good results on longitudinal dynamics control, a more comprehensive NMPC strategy could integrate both longitudinal and lateral motion, enabling full vehicle trajectory planning. This would allow for more complex autonomous driving applications, such as lane-changing maneuvers, obstacle avoidance, and intersection navigation, where both speed and steering must be optimized simultaneously.

Another crucial area to explore is the **robustness of NMPC in real-world driving conditions**. Since the current study was conducted in a controlled simulation environment, real-world factors such as sensor noise, road irregularities, unpredictable traffic behavior, and varying weather conditions should be considered in future studies. Implementing NMPC in an experimental testing ground with real vehicles would provide valuable information into its feasibility outside of simulations.

7.3 Final Remarks

This study demonstrated that Nonlinear Model Predictive Control is a highly effective strategy for improving the efficiency and stability of autonomous electric vehicles.

By exploiting its predictive capabilities, NMPC demonstrated better performance compared to traditional control strategies, particularly in terms of energy efficiency, safety and vehicle stability. The ability to optimize power consumption, limit unnecessary accelerations and adapt to dynamic conditions makes NMPC a highly promising solution for next-generation transportation systems. Its performance in both adaptive cruise control and platoon control scenarios confirms that predictive control strategies can effectively improve the sustainability and safety of autonomous driving.

While further refinements and real-world validation are needed, the results indicate that predictive control strategies like NMPC can play a crucial role in shaping the future of energy-efficient and autonomous mobility. With advancements in computational power and real-time optimization techniques, NMPC has the potential to significantly impact the future of autonomous vehicle control, making autonomous driving not only a reality but also a safer, more efficient, and environmental-friendly solution for modern transportation systems.

Appendix A

MatLab scripts

A.1 NMPC script

```
1
2  clear all
3  close all
4  clc
5
6  addpath ('DrivingCycles');
7  load DrivingCycles\WLTP.mat;
8  addpath ('Battery and EM data');
9  load bat_Ro_vs_SOC_data.mat;
10 load bat_Voc_vs_SOC_data.mat;
11 load BEV_mot.mat;
12 addpath('CasADi');
13 folder='C:\Users\matti\Desktop\MAGISTRALE TORINO\AUTONOMOUS
       VEHICLE\TESI\Result Plots\NMPC';
14
15 % Computation of WLTP Parameters
16 time_WLTP=T_z(1:1800); % WLTP time vector
17 speed_WLTP=V_z(1:1800); % WLTP speed vector
18 pos_WLTP=cumtrapz(time_WLTP, speed_WLTP)+10; % WLTP position
       vector
19 for j=1:length(speed_WLTP)-1
20     acc_WLTP(j,:)=(speed_WLTP(j+1)-speed_WLTP(j))/(time_WLTP(j+1)-
       time_WLTP(j)); % WLTP acceleration vector
21 end
22 acc_WLTP = [acc_WLTP; acc_WLTP(end)]; % Extend to match length to
       match dimensions
23
24 figure()
25 plot(time_WLTP,speed_WLTP,'r');
```

```
26 title('WLTP speed cycle');
27 xlabel('Time [s]')
28 ylabel('Speed [m/s]')
29 legend('WLTP speed cycle')
30 grid on
31
32 %%
33 % List of vehicle parameters
34 r_w=0.3; % Wheel radius [m]
35 M_veh=1400; % Vehicle mass [kg]
36 a=1; % Front axle-CoG [m]
37 b=1.3; % Rear axle-CoG [m]
38 h_CoG=0.3; % Height of CoG [m]
39 grav=9.81; % Gravity [m/s^2]
40 f_0=4.5*1e-3; % Rolling coefficient [N/kN]
41 k=0.013; % Rolling resistance coefficient depending on speed of
       the vehicle [Ns/m]
42 alpha=0; % Road inclination [  ]
43 C_d=0.33; % Drag coefficient
44 A_f=2.15; % Frontal area [m^2]
45 rho= 1.225;% Air density [kg/m^3]
46 tau_gb=9.6; % Gearbox ratio [-]
47 eta_gb=0.97; % Gearbox efficiency [-]
48 Ns=108;% Number of series cells [-]
49 Np=1; % Number of parallel cells [-]
50 Nb=108; % Total number of cells in the battery [-]
51 Qnom=60*Np; % Nominal capacity [Ah]
52 Q_nom=Qnom*3600; % Nominal capacity [As]
53 eta_c=0.95; % Columbic efficiency [-]
54 eta_tr=0.95; % Transmission system efficiency [-]
55 eta_inv=1; % Inverter efficiency [-]
56 eps=0.1;
57
58 T_EM_WLTP=(acc_WLTP*M_veh * r_w)/(eta_gb * tau_gb); % WLTP Torque
       vector
59 w_WLTP= (speed_WLTP*tau_gb/r_w);
60 P_EM_WLTP=T_EM_WLTP.*w_WLTP;
61
62 %starting condition of vehicles
63 x_ego0=0;
64 v_ego0=0;
65 x_lead0=10;
66 v_lead0=0;
67 SOC0=0.8; % Initial SOC [-]
68
69
70 t_sim=1800; % simulation time
71 Td=0.01;
72
```

```
%% NMPC Design
import casadi.*

Ts = 0.5; % Sampling time [s]
N = 5; % prediction horizon (in terms of number of time steps to
    look at)

% Limit values of the state variables
v_max = max(V_z);
v_min = 0;
SOC_max = 1;
SOC_min = 0;
T_EM_max=max(T_EM_WLTP);
T_EM_min=min(T_EM_WLTP);

max_dis=300;
max_dis_urb=100;
max_dis_exurb=200;
max_dis_high=300;
min_dis=20;
max_acc= 1.5;


% State Variables definition
p_veh = SX.sym('p_veh'); % Position
v_veh = SX.sym('v_veh'); % Speed
SOC = SX.sym('SOC');     % State of Charge
states = [ SOC; p_veh; v_veh];
n_states=length(states); % Number of states

% Control Variable
T_EM = SX.sym('T_EM'); % EM Torque
controls=[T_EM];
n_controls=length(controls);

%% SYSTEM MODEL

% Resistive Forces
F_roll = M_veh * grav * f_0; % Rolling resistance
F_aero = 0.5 * rho * C_d * A_f * v_veh^2; % Aerodynamic resistance
F_grade = 0; % Force due to the slope of the street

% Voc and Ro calculation
SOC_table_Voc = SOC_Voc_data(:,1)';
Voc_table = SOC_Voc_data(:,2)';
SOC_table_Ro = SOC_Ro_data(:,1)';
Ro_table = SOC_Ro_data(:,2)';

figure;
```

```
% First Subplot: Voc vs SOC
subplot(2,1,1);
plot(SOC_table_Voc, Voc_table, 'bo-', 'LineWidth', 2, 'MarkerSize', 6);
xlabel('State of Charge (SOC)');
ylabel('Open Circuit Voltage (Voc) [V]');
title('Voc vs SOC');
grid on;

% Second Subplot: Ro vs SOC
subplot(2,1,2);
plot(SOC_table_Ro, Ro_table, 'ro-', 'LineWidth', 2, 'MarkerSize', 6);
xlabel('State of Charge (SOC)');
ylabel('Internal Resistance (Ro) [\Omega]');
title('Ro vs SOC');
grid on;


% Initialize Voc and Ro with zero
Voc = 0;
Ro=0;

% Perform piecewise linear interpolation for Voc
for i = 1:length(SOC_table_Voc)-1
    % Check if SOC is within the interval [SOC_data(i), SOC_data(i+1)]
    in_interval = (SOC >= SOC_table_Voc(i)) & (SOC <= SOC_table_Voc(i+1));

    % Linear interpolation formula
    Voc = Voc + in_interval .* (Voc_table(i) + (Voc_table(i+1) - Voc_table(i)) * (SOC - SOC_table_Voc(i)) / (SOC_table_Voc(i+1) - SOC_table_Voc(i)));
end

% Perform piecewise linear interpolation for Ro
for i = 1:length(SOC_table_Ro)-1
    % Check if SOC is within the interval [SOC_data(i), SOC_data(i+1)]
    in_interval = (SOC >= SOC_table_Ro(i)) & (SOC <= SOC_table_Ro(i+1));

    % Linear interpolation formula
    Ro = Ro + in_interval .* (Ro_table(i) + (Ro_table(i+1) - Ro_table(i)) * (SOC - SOC_table_Ro(i)) / (SOC_table_Ro(i+1) - SOC_table_Ro(i)));
end

```

```
% Values for the entire battery
Vbatt=Voc*Nb;
Rbatt=Ro*Nb;

% Acceleration
a_veh = (T_EM * eta_gb * tau_gb / (M_veh * r_w)) - (F_roll +
    F_aero + F_grade) / M_veh;

% SOC dot calculation
w_EM = v_veh * tau_gb / r_w; % [rad/s]
w_EM_rpm = w_EM * 30 / pi; % [rpm]

% EM Efficiency map
Speed_max = BEV_mot.Speed(:,1);                    % Maximum Speed
    [rpm]
Shaft_Torque_max = BEV_mot.Shaft_Torque(:,1);      % Maximum Torque
     [Nm]

figure
contourf(BEV_mot.Speed,BEV_mot.Shaft_Torque,BEV_mot.Efficiency)
%%
figure;
contourf(BEV_mot.Speed, BEV_mot.Shaft_Torque, BEV_mot.Efficiency,
    20); % Efficiency Map
colorbar;
xlabel('Speed [rpm]');
ylabel('Torque [Nm]');
title('Efficiency Map of the Electric Motor');
grid on;


% 3D EFFICIENCY MAP
figure;
hold on;

h = surf(BEV_mot.Speed, BEV_mot.Shaft_Torque, BEV_mot.Efficiency);
shading interp;
colormap(parula);
h.FaceAlpha = 1;

mesh(BEV_mot.Speed, BEV_mot.Shaft_Torque, BEV_mot.Efficiency, '
    EdgeColor', 'k', 'FaceColor', 'none');

contour3(BEV_mot.Speed, BEV_mot.Shaft_Torque, BEV_mot.Efficiency,
    20, 'k');

hold off;
colorbar;
xlabel('EM speed [rpm]');
```

```
ylabel('Shaft Torque [Nm]');
zlabel('Efficiency [-]');
title('Experimental Efficiency Map');
grid on;
view(-45, 30);


% Look-Up Table Interpolation
speed_values = BEV_mot.Speed(:);
torque_values = BEV_mot.Shaft_Torque(:);
efficiency_values = BEV_mot.Efficiency(:);

speed_axis = unique(BEV_mot.Speed(:,1));
torque_axis_high_res = linspace(min(torque_values), max(
    torque_values), 120); % from 60 to 120 points

% Create the new grid
[Speed_grid_high_res, Torque_grid_high_res] = meshgrid(speed_axis,
     torque_axis_high_res);

% Interpolation on the new grid
Efficiency_grid_high_res = griddata(speed_values, torque_values,
    efficiency_values, Speed_grid_high_res, Torque_grid_high_res, '
    cubic');
Efficiency_grid_high_res=Efficiency_grid_high_res./100;

figure;
subplot(1,2,1);
contourf(BEV_mot.Shaft_Torque, BEV_mot.Speed, BEV_mot.Efficiency);
xlabel('Torque'); ylabel('Speed'); zlabel('Efficiency');
title('Original Efficiency Map');
shading interp;

subplot(1,2,2);
contourf(Torque_grid_high_res, Speed_grid_high_res,
    Efficiency_grid_high_res);
xlabel('Interpolated Torque'); ylabel('Interpolated Speed');
    zlabel('Efficiency');
title('High-Resolution Interpolated Efficiency Map');
shading interp;

% Definition of the dynamic safety distance
reaction_time = 2;     % Reaction time (s)
deceleration = 7;      % Max Deceleration (m/s^2)
braking_distance = v_veh^2 / (2 * deceleration); % Braking
    distance
safety_distance = 1.5*braking_distance + reaction_time * v_veh ; %
     Safety distance
cond=safety_distance<min_dis;
```

```
244 safety_dist=if_else(cond,min_dis,safety_distance);
245 % Symbolic function
246 Safety_Distance_function = Function('Safety_Distance_function', {
    v_veh}, {safety_dist});
247
248
249 % Maximum distance definition
250 cond1= p_veh < 2893.33;
251 cond2= p_veh >= 2893.33 & p_veh < 7850.41;
252 cond3= p_veh >= 7850.41;
253 max_dist=if_else(cond1, max_dis_urb,max_dis);
254 max_dist=if_else(cond2, max_dis_exurb,max_dist);
255 max_dist=if_else(cond3, max_dis_high,max_dist);
256 Max_Distance_function = Function('Max_Distance_function', {p_veh},
     {max_dist});
257
258 %% Efficiency Interpolation
259 % Perform piecewise linear interpolation for T_EM
260     for p = 1:length(Speed_max)-1
261
262         in_int = (w_EM_rpm >= Speed_max(p)) & (w_EM_rpm <=
    Speed_max(p+1));
263
264         % Linear interpolation formula
265         Tem_interp = in_int .* (Shaft_Torque_max(p) + (
    Shaft_Torque_max(p+1) - Shaft_Torque_max(p)) * (w_EM_rpm -
    Speed_max(p)) / (Speed_max(p+1) - Speed_max(p)));
266     end
267
268     Tmax=Tem_interp;
269     Tmin=-Tem_interp;
270
271     % Saturation
272     condition_up = T_EM>Tmax;
273     Tem = if_else(condition_up, Tmax, T_EM);
274     condition_down= T_EM<Tmin;
275     Tem= if_else(condition_down, Tmin, T_EM);
276
277     ind1 = 1;
278     for i = 1:length(torque_axis_high_res)-1
279         condition1 = (Tem >= torque_axis_high_res(i)) & (Tem <
    torque_axis_high_res(i+1));
280         ind1 = if_else(condition1, i, ind1);
281     end
282     ind1 = if_else(Tem >= torque_axis_high_res(end), length(
    torque_axis_high_res) - 1, ind1);
283     ind1 = if_else(Tem < torque_axis_high_res(1), 1, ind1);
284
285     ind2 = 1;
```

```
286     for j = 1:length(speed_axis)-1
287         condition2 = (w_EM_rpm >= speed_axis(j)) & (w_EM_rpm <
    speed_axis(j+1));
288         ind2 = if_else(condition2, j, ind2);
289     end
290     ind2 = if_else(w_EM_rpm >= speed_axis(end), length(speed_axis)
    - 1, ind2);
291     ind2 = if_else(w_EM_rpm < speed_axis(1), 1, ind2);
292 
293     T1 = 0; % Initialize the result
294     for i = 1:length(torque_axis_high_res)-1
295         T1 = T1 + if_else(i == ind1, torque_axis_high_res(i), 0);
296     end
297     T2 = 0; % Initialize the result
298     for j = 1:length(torque_axis_high_res)-1
299         T2 = T2 + if_else(j == ind1, torque_axis_high_res(j+1), 0)
    ;
300     end
301 
302     w1 = 0; % Initialize the result
303     for k = 1:length(speed_axis)-1
304         w1 = w1 + if_else(k == ind2, speed_axis(k), 0);
305     end
306     w2 = 0; % Initialize the result
307     for q = 1:length(speed_axis)-1
308         w2 = w2 + if_else(q == ind2, speed_axis(q+1), 0);
309     end
310 
311     eff11 = 0; % Initialize
312     for i = 1:size(Efficiency_grid_high_res, 1)
313         for j = 1:size(Efficiency_grid_high_res, 2)
314             eff11 = eff11 + if_else((i == ind1) & (j == ind2),
    Efficiency_grid_high_res(i, j), 0);
315         end
316     end
317     eff21 = 0; % Initialize
318     for i = 1:size(Efficiency_grid_high_res, 1)-1
319         for j = 1:size(Efficiency_grid_high_res, 2)-1
320             eff21 = eff21 + if_else((i == ind1) & (j == ind2),
    Efficiency_grid_high_res(i+1, j), 0);
321         end
322     end
323     eff12 = 0; % Initialize
324     for i = 1:size(Efficiency_grid_high_res, 1)-1
325         for j = 1:size(Efficiency_grid_high_res, 2)-1
326             eff12 = eff12 + if_else((i == ind1) & (j == ind2),
    Efficiency_grid_high_res(i, j+1), 0);
327         end
328     end
```

```
329     eff22 = 0; % Initialize
330     for i = 1:size(Efficiency_grid_high_res, 1)-1
331         for j = 1:size(Efficiency_grid_high_res, 2)-1
332             eff22 = eff22 + if_else((i == ind1) & (j == ind2),
    Efficiency_grid_high_res(i+1, j+1), 0);
333         end
334     end
335
336
337     eff_res = 1 / ((T2 - T1) * (w2 - w1)) * ...
338         (eff11 * (T2 - Tem) * (w2 - w_EM_rpm) + ...
339          eff21 * (Tem - T1) * (w2 - w_EM_rpm) + ...
340          eff12 * (T2 - Tem) * (w_EM_rpm - w1) + ...
341          eff22 * (Tem - T1) * (w_EM_rpm - w1));
342
343 eta_EM=eff_res;
344
345 P_b = T_EM * w_EM /((eta_EM * eta_inv)^sign(T_EM*w_EM)); % Battery
     power
346 I_b = (Vbatt-sqrt((Vbatt^2)-4*Rbatt*P_b + eps))/(2*Rbatt); %
    Battery current
347 SOC_dot = (-1/(eta_c^sign(I_b)))*(I_b / Q_nom); % SOC derivative
348
349 % State Equations
350 p_veh_dot = v_veh;
351 v_veh_dot = a_veh;
352
353 % System dynamics
354 sys_dyn = [ SOC_dot; p_veh_dot; v_veh_dot]; % System dynamics
355
356 f = Function('f',{states, controls}, {sys_dyn}); % vehicle
    dynamics function
357
358 U = SX.sym('U',n_controls,N); % Decision variables (controls)
359 X = SX.sym('X',n_states,(N+1)); % Vector that represents the
    states over the optimization problem.
360 P = SX.sym('P',n_states + N*(n_states+n_controls)); % Parameters (
    which include the initial state and the reference along the
361 % predicted trajectory (reference states and reference controls))
362
363 %% COST FUNCTION
364 cost = 0; % Cost function
365 g = [];  % constraints vector
366
367 % Weights for the Cost function
368 lambda1=1;   % weight on P_EM
369 lambda2=1000;    % weight on SOC
370 lambda3=3e4;  % weight on Maximum distance
371 lambda4=100;  % weight on Speed
```

```
372 lambda5 = 500;  % weight on acceleration
373
374
375 st  = X(:,1); % initial state
376 g = [g;st-P(1:3)]; % initial condition constraints
377
378 for k = 1:N
379     st = X(:,k);  % state
380     con = U(:,k); % control
381
382     w_rpm=(X(3,k)*tau_gb/r_w);
383     Tem=U(:,k);
384     Pem=w_rpm*Tem;
385
386     cond_Pem=Pem>0;
387     cond_dist= P((n_states + n_controls) * k + 1) - st(2) >
    Max_Distance_function(st(2));
388
389     cost =  cost + if_else(cond_Pem, 1, 0)*lambda1*(Pem^2); % Cost
     term on the Electric Power
390     cost =  cost + lambda2*(norm(P((n_states+n_controls)*k) - st
    (1)))^2; % Cost term on the SOC
391     cost =  cost + if_else(cond_dist, 1, 0)*lambda3*(norm(P((
    n_states + n_controls) * k + 1) - st(2) - Max_Distance_function
    (st(2))))^2; % Cost term on the Maximum distance
392     cost =  cost + lambda4*(norm(P((n_states+n_controls)*k+2)-st
    (3)))^2 ; % Cost term on the Speed
393
394     st_next = X(:,k+1);
395     f_value = f(st,con);
396     st_next_euler = st+ (Ts*f_value);
397     g = [g;st_next-st_next_euler]; % compute constraints
398 end
399
400 % cost= cost + lambda5*(norm(X(3,N+1)-P((n_states+n_controls)*N+2)
    ))^2; % Cost term on speed at the end of prediction horizon
401
402 for k=1:N-1
403     cost = cost + lambda5 * (norm(X(3,k+1) - X(3,k))/Ts)^2; % Cost
     term on comfort
404 end
405
406
407 for k=1:N
408     speed = X(3,k);  % state
409     g = [g; P((n_states + n_controls) * k + 1) - X(2,k) -
    Safety_Distance_function(speed)]; % Constraint on the minimum
    distance between pos_veh and pos_ref
410 end
```

```
411
412
413 % make the decision variable one column  vector
414 OPT_variables = [reshape(X,3*(N+1),1);reshape(U,N,1)];
415
416 nlp_prob = struct('f', cost, 'x', OPT_variables, 'g', g, 'p', P);
417
418 opts = struct;
419 opts.ipopt.max_iter = 10000;
420 opts.ipopt.print_level =0;%0,3
421 opts.print_time = 0;
422 opts.ipopt.acceptable_tol =1e-8;
423 opts.ipopt.acceptable_obj_change_tol = 1e-6;
424
425 solver = nlpsol('solver', 'ipopt', nlp_prob,opts);
426
427 %% CONSTRAINTS
428 args = struct;
429
430 % Constraint for the system to respect the vehicle dynamics
    function
431 args.lbg(1:3*(N+1)) = 0;  % -1e-20  % Equality constraints
432 args.ubg(1:3*(N+1)) = 0;  % 1e-20   % Equality constraints
433
434 args.lbg(3*(N+1)+1:3*(N+1)+N) = 0;  % Constraint on the minimum
    distance
435 args.ubg(3*(N+1)+1:3*(N+1)+N) = inf;  % Constraint on the minimum
    distance
436
437 % Constraints on state variables
438 args.lbx(1:3:3*(N+1),1) = SOC_min; % SOC lower bound
439 args.ubx(1:3:3*(N+1),1) = SOC_max; % SOC upper bound
440 args.lbx(2:3:3*(N+1),1) = 0; % Position lower bound
441 args.ubx(2:3:3*(N+1),1) = inf; % Position upper bound
442 args.lbx(3:3:3*(N+1),1) = v_min; % Speed lower bound
443 args.ubx(3:3:3*(N+1),1) = v_max; % Speed upper bound
444
445 % Constraints on the control variable
446 args.lbx(3*(N+1)+1:1:3*(N+1)+N,1) = T_EM_min; % T_EM lower bound
447 args.ubx(3*(N+1)+1:1:3*(N+1)+N,1) = T_EM_max; % T_EM upper bound
448
449
450 % Interpolate WLTP speed profile to match simulation time
451 time_sim = linspace(time_WLTP(1), time_WLTP(end), t_sim / Ts); %
    Create simulation time vector
452 speed_ref = interp1(time_WLTP, speed_WLTP, time_sim, 'linear'); %
    Interpolate
453 pos_ref = interp1(time_WLTP, pos_WLTP, time_sim, 'linear'); %
    Interpolate
```

```
454 acc_ref = interp1(time_WLTP, acc_WLTP, time_sim, 'linear'); %
    Interpolate
455 T_EM_ref= interp1(time_WLTP, T_EM_WLTP, time_sim, 'linear'); %
    Interpolate
456 P_EM_ref= interp1(time_WLTP, P_EM_WLTP, time_sim, 'linear'); %
    Interpolate
457
458 speed_ref(end:end+N)=speed_ref(end);
459 pos_ref(end:end+N)=pos_ref(end);
460 T_EM_ref(end:end+N)=T_EM_ref(end);
461
462 %---------------------------------------------
463 % ALL OF THE ABOVE IS JUST A PROBLEM SET UP
464 %%
465
466 % THE SIMULATION LOOP STARTS FROM HERE
467 %-----------------------------------------
468 t0 = 0;
469 x0 = [SOC0 ; 0 ; 0];     % initial condition
470
471 xx(:,1) = x0; % xx contains the history of states
472 t(1) = t0; % t contains the time
473
474 u0 = zeros(N,1);         % control inputs over the prediction time
475 X0 = repmat(x0,1,N+1)'; % initialization of the states decision
    variables
476
477 sim_tim = 1800; % Maximum simulation time
478
479 % Start MPC
480 iter_number = 0; % Iteration counter
481 xx1 = [];
482 u_cl=[]; % Control variable over the entire cycle
483
484 reference=zeros(4,sim_tim); % reference vector initialization ( 4
    = [SOC_ref, pos_ref, speed_ref, T_EM_ref])
485 main_loop = tic;
486
487 % the main simulaton loop... it works as long the number of NMPC
    steps is less than its maximum value.
488
489 while(iter_number < sim_tim/Ts) % Condition for ending the loop
490     current_time = iter_number;  % get the current time
491     %
    ------------------------------------------------------------------------

492     args.p(1:3) = x0; % initial condition of the state
493     for k = 1:N % set the reference to track
494         t_predict = current_time + k; % predicted time instant
```

```
495
496         SOC_ref=SOC0; % our reference in terms of SOC is to keep
    SOC equal to its initial value (0.8)
497         x_ref = pos_ref(t_predict);
498         v_ref = speed_ref(current_time+1); % The reference speed
    remains constant at the starting value during the prediction
499         Tem_ref = T_EM_ref(t_predict);
500
501         args.p((n_states+n_controls)*k:(n_states+n_controls)*k+2)
    = [SOC_ref, x_ref, v_ref];
502         args.p((n_states+n_controls)*k+n_states) = [Tem_ref];
503
504         reference(1,current_time+1)=SOC_ref;
505         reference(2,current_time+1)=x_ref;
506         reference(3,current_time+1)=v_ref;
507         reference(4,current_time+1)=Tem_ref;
508
509     end
510     %
    ---------------------------------------------------------------------

511     % initial value of the optimization variables
512     args.x0  = [reshape(X0',3*(N+1),1);reshape(u0',N,1)];
513     sol = solver('x0', args.x0, 'lbx', args.lbx, 'ubx', args.ubx
    ,...
514         'lbg', args.lbg, 'ubg', args.ubg,'p',args.p);
515     u = reshape(full(sol.x(3*(N+1)+1:end))',1,N)'; % get controls
    from the solution
516     xx1(:,1:3,iter_number+1)= reshape(full(sol.x(1:3*(N+1)))',3,N
    +1)'; % get solution TRAJECTORY
517     u_cl= [u_cl ; u(1,:)];
518     t(iter_number+1) = t0;
519     % Apply the control and shift the solution
520     [t0, x0, u0] = shift(Ts, t0, x0, u,f);
521     xx(:,iter_number+2) = x0;
522     X0 = reshape(full(sol.x(1:3*(N+1)))',3,N+1)'; % get solution
    TRAJECTORY
523     % Shift trajectory to initialize the next step
524     X0 = [X0(2:end,:);X0(end,:)];
525     iter_number
526     iter_number = iter_number + 1;
527 end;
528
529
530 main_loop_time = toc(main_loop);
531 average_mpc_time = main_loop_time/(iter_number+1)
532 SOC_final=xx(1,end)
533 %%
534 % Acceleration computation
```

```
535 ref_speed=reference(3,1:sim_tim/Ts);
536 for j=1:length(ref_speed)-1
537     acc(j)=(ref_speed(j+1)-ref_speed(j))/(time_sim(j+1)-time_sim(j
    )); % acceleration vector
538 end
539 acc_ref = [acc acc(end)]; % Extend to match length to match
    dimensions
540
541 speed_vect=xx(3,1:sim_tim/Ts);
542 for j=1:length(speed_vect)-1
543     acc_vec(j)=(speed_vect(j+1)-speed_vect(j))/(time_sim(j+1)-
    time_sim(j)); % acceleration vector
544 end
545 acc_vect = [acc_vec acc_vec(end)]; % Extend to match length to
    match dimensions
546 %%
547 % Total power provided by the battery
548 v_cl=xx(3,1:end-1);
549 w_cl=(((v_cl .* tau_gb) ./ r_w)); % [rad/s]
550 T_EM_cl=u_cl';
551 P_EM_cl = T_EM_cl .* w_cl; % .//((eta_EM .* eta_inv).^sign(T_EM_cl
    .*w_rpm_cl));
552
553 total_P_EM=trapz(time_sim, P_EM_cl);
554 disp(['The total power provided by the EM is: ' num2str(total_P_EM
    )]);
555 %%
556 for i=10:length(time_sim)
557     if xx(2,i+1)>= reference(2,i)
558
559         disp(['THERE IS A CRASH at instant ' num2str(i)]);
560     end
561 end
562
563 %% MAX DISTANCE OVER THE SIMULATION
564 dist_max=0;
565 for i=1:length(time_sim)
566     if reference(2,i)-xx(2,i)>= dist_max;
567     dist_max = reference(2,i)-xx(2,i);
568     end
569 end
570 dist_max
571
572 %% RESULTS PLOT
573 folder='C:\Users\matti\Desktop\MAGISTRALE TORINO\AUTONOMOUS
    VEHICLE\TESI\Result Plots\NMPC';
574
575 % Plotting the SOC of the vehicle
576 figure()
```

```
577 plot(time_sim,reference(1,1:sim_tim/Ts),'b',time_sim,xx(1,1:
        sim_tim/Ts),'r');
578 title('SOC of the vehicle controlled by NMPC');
579 xlabel('Time [s]')
580 ylabel('SOC [-]')
581 legend('Reference SOC','SOC')
582 ylim([0.7,0.82]);
583 grid on
584 saveas(gcf, fullfile(folder, 'SOC_NMPC.png'));
585
586 % Plotting the position of the vehicle
587 figure()
588 plot(time_sim,reference(2,1:sim_tim/Ts),'b',time_sim,xx(2,1:
        sim_tim/Ts),'r');
589 title('Position of the two vehicles in NMPC simulation');
590 xlabel('Time [s]')
591 ylabel('Position [m]')
592 legend('Reference Position','Actual Position')
593 grid on
594 saveas(gcf, fullfile(folder, 'Position_NMPC.png'));
595
596 % Plotting the speed of the vehicle
597 figure()
598 plot(time_sim,reference(3,1:sim_tim/Ts),'b',time_sim,xx(3,1:
        sim_tim/Ts),'r');
599 title('Speed of the two vehicles in NMPC simulation');
600 xlabel('Time [s]')
601 ylabel('Speed [m/s]')
602 legend('Reference Speed','Actual Speed')
603 grid on
604 saveas(gcf, fullfile(folder, 'Speed_NMPC.png'));
605
606 % Plotting the acceleration of the vehicle
607 figure()
608 plot(time_sim,acc_ref,'b',time_sim,acc_vect,'r');
609 title('Acceleration of the two vehicles in NMPC simulation');
610 xlabel('Time [s]')
611 ylabel('Acceleration [m/(s^2)]')
612 legend('Reference Acceleration','Actual Acceleration')
613 grid on
614 saveas(gcf, fullfile(folder, 'Acceleration_NMPC.png'));
615
616 u_cl=u_cl';
617 % Plotting the T_EM of the vehicle
618 figure()
619 plot(time_sim,reference(4,1:sim_tim/Ts),'b',time_sim,u_cl(1,1:(
        sim_tim/Ts)),'r');
620 title('EM Torque of the two vehicles in NMPC simulation');
621 xlabel('Time [s]')
```

```
622 ylabel('T_EM [Nm]')
623 legend('Reference EM Torque','Commanded EM Torque')
624 grid on
625 saveas(gcf, fullfile(folder, 'TEM_NMPC.png'));
626
627 % Plotting the P_EM of the vehicle
628 figure()
629 plot(time_sim, P_EM_ref,'b',time_sim, P_EM_cl(:,1:sim_tim/Ts),'r')
    ;
630 title('EM Power of the two vehicles in NMPC simulation');
631 xlabel('Time [s]')
632 ylabel('P_EM ')
633 legend('EM Power of WLTP cycle','EM Power provided by Vehicle')
634 grid on
635 saveas(gcf, fullfile(folder, 'PEM_NMPC.png'));
636
637 %% RESULTS SAVING
638 folder='C:\Users\matti\Desktop\MAGISTRALE TORINO\AUTONOMOUS
    VEHICLE\TESI\Platoon results';
639
640 % Saving WLTP results for plotting
641 time_wltp=time_sim;
642 soc_wltp=reference(1,1:sim_tim/Ts);
643 position_wltp=reference(2,1:sim_tim/Ts);
644 vel_wltp=reference(3,1:sim_tim/Ts);
645 accel_wltp=acc_ref;
646 tem_wltp=reference(4,1:sim_tim/Ts);
647 pem_wltp=P_EM_ref;
648
649 save(fullfile(folder, 'WLTP_ref.mat'), 'time_wltp', 'soc_wltp', '
    position_wltp', 'vel_wltp', 'accel_wltp', 'tem_wltp', 'pem_wltp
    ');
650
651 % Saving vehicle results for plotting
652 time=time_sim;
653 soc=xx(1,1:sim_tim/Ts);
654 position=xx(2,1:sim_tim/Ts);
655 vel=xx(3,1:sim_tim/Ts);
656 accel=acc_vect;
657 tem=u_cl(1,1:(sim_tim/Ts));
658 pem=P_EM_cl(:,1:sim_tim/Ts);
659 soc_final=SOC_final;
660 pem_total=total_P_EM;
661
662 save(fullfile(folder, 'veh1.mat'), 'time', 'soc', 'position', 'vel
    ', 'accel', 'tem', 'pem', 'soc_final','pem_total');
663
664 %%
```

```
665 folder='C:\Users\matti\Desktop\MAGISTRALE TORINO\AUTONOMOUS
    VEHICLE\TESI\Result Plots\NMPC';
666
667 addpath("CTG");
668 final_SOC_CTG=load("CTG\final_SOC_CTG.mat");
669 total_PEM_CTG=load("CTG\total_PEM_CTG.mat");
670 final_SOC_CTG=final_SOC_CTG.SOC_final;
671 total_PEM_CTG=total_PEM_CTG.P_EM_tot_ego;
672
673 final_SOC_vect=[final_SOC_CTG; SOC_final];
674 % Plotting the final value of SOC
675 figure;
676 hold on;
677 num = size(final_SOC_vect, 1);
678 colors = [1 0 0;  % red
679           0 0 1];  % blue
680 b = gobjects(num,1);
681 for i = 1:num
682     b(i) = bar(i, final_SOC_vect(i), 'FaceColor', colors(i,:), '
    EdgeColor', 'k');
683 end
684 title('Comparison on the final SOC value');
685 xticks(1:2);
686 xticklabels({'CTG', 'NMPC'});
687 ylabel('SOC [-]');
688 ylim([0.6, 0.8]);
689 grid on;
690 saveas(gcf, fullfile(folder, 'SOC_final.png'));
691 %%
692 total_PEM_vect=[total_PEM_CTG; total_P_EM];
693 % Plotting the total value of P_EM
694 figure;
695 hold on;
696 num_ = size(total_PEM_vect, 1);
697 colors = [1 0 0;  % red
698           0 0 1];  % blue
699 b = gobjects(num_,1);
700 for i = 1:num_
701     b(i) = bar(i, total_PEM_vect(i), 'FaceColor', colors(i,:), '
    EdgeColor', 'k');
702 end
703 title({'Comparison on the total value of Power'},{'\bf provided by
     EM during the whole simulation'});
704 xticks(1:2);
705 xticklabels({'CTG', 'NMPC'});
706 ylabel('P_{EM} [W]');
707 ylim([0, 5e7])
708 grid on;
709 % saveas(gcf, fullfile(folder, 'total_PEM.png'));
```

```
710
711
712 %%
713 % SHIFT FUNCTION
714 function [t0, x0, u0] = shift(Ts, t0, x0, u,f)
715 st = x0;
716 con = u(1,:)';
717 f_value = f(st,con);
718 st = st+ (Ts*f_value);
719 x0 = full(st);
720
721 t0 = t0 + Ts;
722 u0 = [u(2:size(u,1),:);u(size(u,1),:)];
723 end
```

Bibliography

[1] EC-European Commission et al. «Regulation (EU) 2019/631 of the European Parliament and of the Council of 17 April 2019 setting CO2 emission performance standards for new passenger cars and for new light commercial vehicles, and repealing Regulations (EC) No 443/2009 and (EU) No 510/2011 (recast)». In: *Official Journal of the European Union* L 111 (2019), pp. 13–53 (cit. on p. 2).

[2] Haohua Zhang. «Enhancing vehicle safety - the role of PID control in adaptive cruise control systems». In: *AIP Conference Proceedings* 3194.1 (Dec. 2024), p. 050023. DOI: 10.1063/5.0193921 (cit. on p. 7).

[3] Harshit Jain and Priyal Babel. «A Comprehensive Survey of PID and Pure Pursuit Control Algorithms for Autonomous Vehicle Navigation». In: (2024). DOI: 10.48550/arXiv.2409.09848. arXiv: 2409.09848 [cs.RO] (cit. on p. 7).

[4] Abdussalam Ali Ahmed, Mohamed Almihat, Abdulgader Alsharif, Mohamed Belrzaeg, and Mohamed Khaleel. «LINEAR QUADRATIC REGULATOR CONTROLLER (LQR) FOR VEHICLE YAW RATE EVALUATION». In: Dec. 2022 (cit. on p. 7).

[5] Tuan Anh Nguyen. «Control an Active Suspension System by Using PID and LQR Controller». In: *International Journal of Mechanical and Production Engineering Research and Development* 10 (Jan. 2020), pp. 7003–7012. DOI: 10.24247/ijmperdjun2020662 (cit. on p. 7).

[6] Ning Ye, Duo Wang, and Yong Dai. «Enhancing Autonomous Vehicle Lateral Control: A Linear Complementarity Model-Predictive Control Approach». In: *Applied Sciences* 13 (Sept. 2023), p. 10809. DOI: 10.3390/app131910809 (cit. on p. 7).

[7] Paolo Falcone, Francesco Borrelli, Jahan Asgari, Eric Tseng, and Davor Hrovat. «Predictive Active Steering Control for Autonomous Vehicle Systems». In: *Control Systems Technology, IEEE Transactions on* 15 (June 2007), pp. 566–580. DOI: 10.1109/TCST.2007.894653 (cit. on p. 7).

[8] Y. Gao and F. Borrelli. «Model Predictive Control for Autonomous Driving: Multi-Objective Optimization and Control Architecture». In: (2010), pp. 205–210 (cit. on p. 7).

[9] Paolo Falcone, Francesco Borrelli, Jahan Asgari, Eric Tseng, and Davor Hrovat. «Predictive Active Steering Control for Autonomous Vehicle Systems». In: *Control Systems Technology, IEEE Transactions on* 15 (June 2007), pp. 566–580. DOI: 10.1109/TCST.2007.894653 (cit. on p. 8).

[10] Stefano Di Cairano, Eric Tseng, Daniele Bernardini, and Alberto Bemporad. «Vehicle Yaw Stability Control by Coordinated Active Front Steering and Differential Braking in the Tire Sideslip Angles Domain». In: *IEEE Trans. Control Systems Technology* 21 (June 2013), pp. 1236–1248. DOI: 10.1109/TCST.2012.2198886 (cit. on p. 8).

[11] Ugo Rosolia and Francesco Borrelli. «Learning Model Predictive Control for Iterative Tasks. A Data-Driven Control Framework». In: *IEEE Transactions on Automatic Control* PP (Sept. 2017), pp. 1–1. DOI: 10.1109/TAC.2017.2753460 (cit. on p. 8).

[12] Matthew Brown, Joseph Funke, Stephen Erlien, and J. Gerdes. «Safe driving envelopes for path tracking in autonomous vehicles». In: *Control Engineering Practice* 61 (May 2016). DOI: 10.1016/j.conengprac.2016.04.013 (cit. on p. 8).

[13] V. Hermansson and K. Moparthi. «Control of an electric vehicle powertrain to mitigate shunt and shuffle». In: (2016) (cit. on p. 13).

[14] Hicham El Hadraoui, Mourad Zegrari, Ahmed Chebak, Oussama Laayati, and Nasr Guennouni. «A Multi-Criteria Analysis and Trends of Electric Motors for Electric Vehicles». In: *World Electric Vehicle Journal* (2022). URL: https://api.semanticscholar.org/CorpusID:248057564 (cit. on p. 14).

[15] Lorenzo Calogero, Michele Pagone, Francesco Cianflone, Edoardo Gandino, Carlo Karam, and Alessandro Rizzo. «Neural Adaptive MPC with Online Metaheuristic Tuning for Power Management in Fuel Cell Hybrid Electric Vehicles». In: *IEEE Transactions on Automation Science and Engineering* (2025), pp. 1–1. DOI: 10.1109/TASE.2025.3534402 (cit. on p. 16).

[16] S. Onori, L. Serrao, and G. Rizzoni. *Hybrid Electric Vehicles: Energy Management Strategies*. Springer, 2016 (cit. on p. 16).

[17] Raffaele Manca, Eugenio Tramacere, Stefano Favelli, Andrea Tonoli, and Luca Camosi. «Impact of Rolling Resistance Modeling on Energy Consumption for a Low Voltage Battery Electric Vehicle». In: *2023 International Symposium on Electromobility (ISEM)*. IEEE, 2023, pp. 1–6 (cit. on p. 21).

[18] Johannes Köhler, Elisa Andina, Raffaele Soloperto, Matthias A. Müller, and Frank Allgöwer. «Linear Robust Adaptive Model Predictive Control: Computational Complexity and Conservatism». In: *2019 IEEE Conference on Decision and Control (CDC)*. Dec. 2019. DOI: 10.1109/cdc40024.2019.9028970. URL: https://scite.ai/reports/10.1109/cdc40024.2019.9028970 (cit. on p. 22).

[19] Rajesh Rajamani. *Vehicle dynamics and control*. Springer Science & Business Media, 2011 (cit. on p. 33).

[20] Carlos Flores, Vicente Milanés, and Fawzi Nashashibi. «A Time Gap-Based Spacing Policy for Full-Range Car-Following». In: *Intelligent Transportation Systems Conference 2017*. Yokohama, Japan, Oct. 2017 (cit. on p. 37).

Acknowledgements

At the conclusion of this thesis, and in general of this academic path, I would like to dedicate a space in my thesis to express my gratitude to all those who have supported me throughout this journey. Their guidance, encouragement, and presence have been invaluable, and I deeply appreciate their contributions to my academic and personal growth.

I would like to express my deepest gratitude to my professors, whose support and guidance have been fundamental throughout my academic journey and have contributed to my growth—not only academically but also personally and professionally.

In particular, a special thanks goes to my thesis supervisors, professor Angelo Bonfitto and professor Michele Pagone, for their availability, invaluable advice, and constant encouragement, which have been essential in the development of this thesis.

I am also grateful to Politecnico di Torino for the numerous learning and growth opportunities it has provided me over these two years. Thanks to the experiences I have had, the courses I have attended, and the people I have met, I have been able to expand my knowledge and develop new skills, that will help me to approach my professional future with greater awareness.

A thank you goes also to my colleagues and friends, who have shared this academic journey with me. Their support, collaboration, and encouragement have made this experience even more enriching. I am grateful for the friendships built along the way and for the motivation we have given each other to face challenges together.

Lastly, a heartfelt thank you to everyone who, in one way or another, has made this journey so enriching and meaningful.